2014

ANALYSE AV BANEGENERATOR

MASTEROPPGAVE
I ELEKTRONIKK OG DATATEKNOLOGI
FYSISK INSTITUTT
DET MATEMATISK-NATURVITENSKAPELIGE FAKULTET

ALYA HOSSAIN

Forord

Denne oppgaven er formulert og gitt av Universitetet senteret på Kjeller (UNIK) i vårsemesteret 2014. Oppgaven er en avsluttende del av masterutdanningen i elektronikk og datateknologi ved Universitetet i Oslo (UIO), utført ved UNIK. Prosjektet gjennomføres av undertegnende, som vil takke intern veileder og faglærer Oddvar Hallingstad for stor hjelp og verdifull kunnskap som har gjort oppgaven veldig lærerikt.

Alya Hossain

Universitetet i Oslo

Sammendrag

Oppgaven går ut på å analysere en banegenerator for et fly eller undervannsfarkost som er representert i planet. Målet er å finne ut en algoritme som gir ut sanne tilstander. Tilstandene beskriver fartøyet posisjon, hastighet og akselerasjon. Kravet er at disse tilstandene skal være kontinuerlige.

Dette har blitt gjort ved å lage en matematisk modell av bane som baseres på rett og sirkel bevegelse. For overgang fra rett strekning til en sirkel ble det bruket Eulerspiral. I tillegg har det blitt testet forskjellige algoritmer ved bruk av spline metode. Banen som er representert fra spline metoden vil diskuteres og sammenliknes med banen som er bygd av rettstrekning, Eulerspiral og sirkel.

Innholdsfortegnelse

Forord I

Sammendrag II

Figurliste V

1. Innledning 1

1.1 Generelt 1

1.2 Målsetting for oppgaven 1

1.3 Kapittelinndeling 2

2. Matematisk grunnlag 3

2.1 Vektor og matriser 3

2.2 Rotasjonskosinmatrise (RKM) 4

2.3 Eulervinkler 4

2.4 Koordinatsystemer og rammer 5

3 Spline metode 6

3.1 Definisjon 6

3.2 Egenskaper 8

3.3.1 Bezier Spline 10

3.3.3 Kubisk Hermite spline 12

3.3.1 B-Splines 13

3.3.4 Naturlig Kubisk Spline 15

4 Banegenerator 17

4.1 Banegenerator basert på rettlinje og sirkelbue 18

4.1.1 Rett bane: 18

4.1.2 Eulerspiral 19

4.1.3 Sirkelbane 22

4.1.4 Omvendt Eulerspiral 23

4.2 Banegenerator basert på Spline metoden 25

4.2.1 Polynombane 25

4.2.2 Splinebane 27

4.3 Beskrivelse av MATLAB programmet 33

4.3.1 Sammensattbane 33

4.3.2 *RBA*.m.funksjon 35

4.3.3 ES.m funksjon 36

4.3.4 SB.m funksjon 37
4.3.5 OES.m 38
4.3.6 PB.m funksjon 39
4.3.7 SP3B.m funksjon 41
4.3.8 SP4B.m funksjon 42
4.3.9 SP5B.m funksjon 43
5. Simuleringsresultater 44
5.1 Eulerspiral 45
5.2 Bane i svingebevegelse 49
5.3 Sammensatt av baneelementer 52
5.4 bane i sirkelbevegelse 57
6 Konklusjon og videre arbeid 59
6.1 Konklusjon 59
6.2 Videre arbeid 60
Referanser 61
Vedlegg 62
A. Funksjon 62
A.1 RBA.m 62
A.2 SB.m 62
A.3 ES.m 62
A.4 OES.m 62
A.5 PB.m 62
A.6 SP3B.m 63
A.7 SP4B.m 64
A.8 SP5B.m 65
B. Simulering 66

Figurliste

Figur (2-1) Eulervinkler [1] 5
Figur (3-1) $C0$ Kontinuitet mellom to kurvesegmenter 8
Figur (3-2) $C1$Kontinuitet mellom to kurvesegmenter 8
Figur (3-3) $C2$Kontinuitet mellom to kurvesegmenter 9
Figur (3-4) Kubikk Bezier Spline [3] 11
Figur (3-5) B-Spline kurv 13
Figur (3-6) naturlig kubisk spline [3] 15
Figur (4-1) Sammensatt baneelementer 17
Figur (4-2) Eulerspiral med θverdier 19
Figur (4-3) RBA.m funksjon 35
Figur (4-4) ES.m funksjon 36
Figur (4-5) SB.m funksjon 37
Figur (4-6) PB.m funksjon 40
Figur (5-1) sammensattbane (Eulerspiral) 45
Figur (5-2) 3.gradsnaturlig splinebane (sammenliknet med Eulerspiral) 46
Figur (5-3) 4.gradsnaturlig splinebane (sammenliknet med Eulerspiral) 47
Figur (5-4) 5.gradsnaturlig splinebane (sammenliknet med Eulerspiral) 48
Figur (5-5) Sammensattbane i svingebevegelse 49
Figur (5-6) 4.gradsnaturlig splinebane i svingebevegelse 50
Figur (5-7) 5.gradsnaturlig splinebane i svingebevegelse 51
Figur (5-8) Sammensattbane 52
Figur (5-9) 4.gradsnaturlig splinebane 53
Figur (5.10) differanse mellom referanse bane og 4.grads naturlig spline bane 54
Figur (5-11) 5.gradsnaturlig splinebane 55
Figur (5.12) differanse mellom referanse bane og 5.grads naturlig spline bane 56
Figur (5-13) bane i sirkelbevegelse 57
Figur (5.14) 5.gradsnaturlig spline bane i sirkelbevegelse 58

1. Innledning

1.1 Generelt

Oppgaven er gitt av UNIK universitetet senter på kjeller med intern veileder Oddvar Hallingstad. I denne forbindelse vil emnet UNIK 4540-Matematisk modellering av dynamiske systemer og emnet MAT-INF4170 – Spline metoder være med å støtte opp mye av den matematiske teorien i rapporten. Mye matematikk og studering av nye teori, har ført til at oppgaven har vært veldig lærerik.
Oppgaven ble utdelt 20. januar 2014 med en tidsramme på 18 uker som leveres innen 26. mai 2014.

1.2 Målsetting for oppgaven

Målet for oppgave er å lage en banegenerator for et fly, båt eller undervannsfarkost som kan brukes til å analysere treghetsnavigasjonssystemet.

For å realisere en banegenerator skal følgende krav oppfølges:

1. Lage en algoritme som vil gi kontinuerlig hastighet og akselerasjon i 2-dimensjonalt affint rom basert på forhåndsdefinerte baneelementer.
2. Undersøke en algoritme som vil gi kontinuerlig hastighet og akselerasjon basert på spline metode. Banen skal passere gjennom forhåndsdefinert veipunkter.
3. Implementere de valgte algoritmer i MATLAB programmet og sammenligne metodene.

1.3 Kapittelinndeling

Kapittel 2: Matematisk grunnlag
I dette kapittel beskrives ulike matematiske grunnlag for oppgaven som er tatt opp i forskjellige deler.

Kapittel 3: Spline metoden
I dette kapittel beskrives forskjellige type av Spline metoder som er undersøkt til å definere en banegenerator.

Kapittel 4: Banegenerator
I dette kapittel beskrives en matematiske fremgangsmåte for å analysere en banegenerator. Tilslutt beskrives funksjoner for programmeringen som er utført.

Kapittel 5: Simuleringsresultater
I dette kapittel blir resultatene som er oppnådd fremstilt i grafer og figurer.

Kapittel 6: Konklusjon og videre arbeid
I dette kapittel blir oppnådde resultater oppsummert og tatt opp forslag til videre arbeid for oppgaven som er basert på de oppnådde resultater.

2. Matematisk grunnlag

I denne delen blir ulike matematiske grunnlag for oppgaven tatt opp i forskjellige deler. Dette viser begrunnelse for valg av metoder og framgang i matematikken og hvorfor bestemte algoritmer og beregninger er blitt valgt til løsning.

2.1 Vektor og matriser

Vektor og matriseberegninger brukes generelt for å finne avstander, retninger og oppførselen til et legemet når man ser bort fra fysiske krefter som har en påvirkning. Om legemet er i lufta, på bakken eller under vann har ingen betydning, samme matematiske formler kan benyttes hvor fysiske lover er relevant.

En type vektor blir definert i et vektor rom eller et affint rom. Notasjonen for de sentrale begrepene i et vektor og affint rom innenfor matematikken er gitt i tabellen [1].

Rom	Rammer	Andre navn på rammer
$\mathcal{V}$: vektorrom	$F_a^{\mathcal{V}} = \{\vec{a}_1, \vec{a}_2, \vec{a}_3\}$: ramme a i vektorrom $\mathcal{V}$ med basisvektorer $\vec{a}_i$	basis, basissystem, basisvektorsett k.s.

Tabell 1: Notasjon for vektorrom

Rom	Rammer	Andre navn på rammer
$\mathcal{A}$: affint rom	$F_a^{\mathcal{A}} = \{O_a, \vec{a}_1, \vec{a}_2, \vec{a}_3\} = \{O_a, \vec{x}_a, \vec{y}_a, \vec{z}_a\}$ ramme a i affint rom A med Origo O_a og basisvektorer $\vec{a}_i$	k.s.

Tabell 2: Notasjon for affint rom

2.2 Rotasjonskosinmatrise (RKM)

Retningskosinmatrise (RKM) brukes her for å finne en overgang fra to eller flere forskjellige referanserammer i et helt system. Det kan forklares ved å anta en ramme a og en ramme b. I dette tilfelle defineres RKM som en stillingsmatrise som gir stilling (orienteringen) av ramme *b* i ramme *a*: dvs. kolonnene i R_b^a er en kolonnerepresentasjon av *b*-basisen i *a*-basisen [1].

$$\vec{b}_i = R_a^b \, \vec{a}_i \tag{2.1}$$

Hvor $i = 1,2,3, \ldots.$

Hvor basisvektorsettet $\{\vec{a}_i\}$ roteres til en ny stilling slik at $\vec{a}_i \rightarrow \vec{b}_i$

$$R_b^a = [\underline{b}_1^a \, ; \, \underline{b}_2^a ; \, \underline{b}_3^a] \tag{2.2}$$

2.3 Eulervinkler

Eulervinkler [1] er en metode som benyttes for å beskrive stillingsformen til et legeme. I en bestemt ramme som er definert, roteres tre forskjellige koordinatakser med vinklene $(\theta_1, \theta_2, \theta_3)$. Disse vinklene roteres om x, y og z aksene . Aksene står ortonormale på hverandre og oppfyller høyrehåndsregelen. Ved å multiplisere disse tre rotasjonene kan man finne rotasjonen mellom to forskjellige rammer [1].

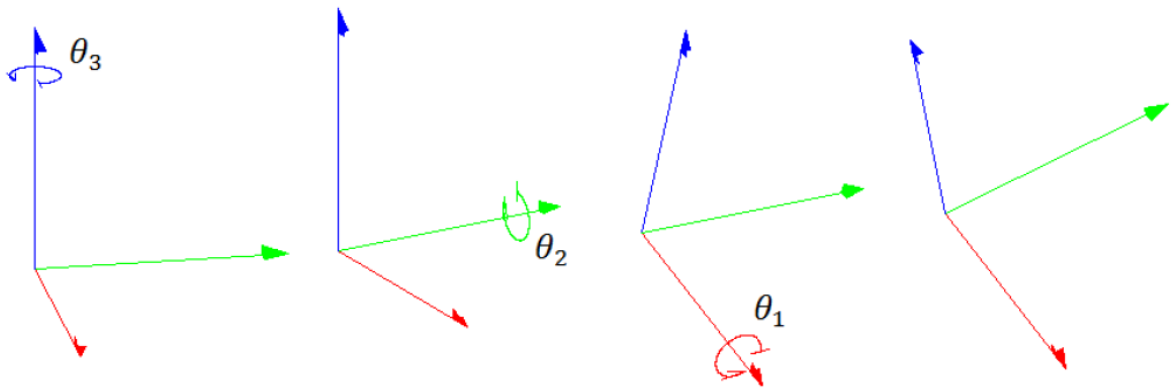
$$R_b^a \, (\theta_1, \theta_2, \theta_3) \; = \; R(\theta_3) \, R(\theta_2) R(\theta_1) \tag{2.3}$$

De tre generelle rotasjonsmatrise til Euler definert som:

$$R(\theta_3, z) = \begin{bmatrix} cos\theta_3 & -sin\theta_3 & 0 \\ sin\theta_3 & cos\theta_3 & 0 \\ 0 & 0 & 0 \end{bmatrix} \tag{2.4}$$

$$R(\theta_2, y) = \begin{bmatrix} cos\theta_2 & 0 & sin\theta_2 \\ 0 & 0 & 0 \\ -sin\theta_2 & 0 & cos\theta_2 \end{bmatrix} \tag{2.5}$$

$$R(\theta_1, x) = \begin{bmatrix} 0 & 0 & 0 \\ 0 & cos\theta_1 & -sin\theta_1 \\ 0 & sin\theta_1 & cos\theta_1 \end{bmatrix} \tag{2.6}$$



Figur (2-1) Eulervinkler [1]

2.4 Koordinatsystemer og rammer

Når man foretar måling eller beregninger av et fartøys bevegelse, refereres dette i forhold til ulike referansesystemer eller rammer. I dette tilfelle vil ortonormale rammer være:
$f^n = \{0_n; \vec{n}_1, \vec{n}_2, \vec{n}_3\}$ Global ramme for banen

$f^a = \{0_a; \vec{a}_1, \vec{a}_2, \vec{a}_3\}$ Lokal ramme for baneelementer f^astår i ro i forhold til f^n. Bevegelsen starter langs $\vec{a}_1 - aksen$ i 0_a
$f^b = \{0_b; \vec{b}_1, \vec{b}_2, \vec{b}_3\}$ Lokal ramme for baneelementer hvor $\vec{b}_1$peker langs hastighetsvektoren for punktet og $\vec{b}_2$ peker mot venstre

3 Spline metode

I dette kapittel vil det undersøkes om Spline metoden kan brukes til å definere banen til et fly eller undervannsfarkost som beveger seg gjennom veldefinert veipunkter i 2-dimensjonalt koordinat systemet. Gjennom dette kapitlet setter vi fokus på anvendelse av kurvetilpasning slik at vi får en jevn kurv mellom veipunker.

3.1 Definisjon

Spilne kurve er en funksjon av stykkevis polynomer hvor polynom likningen er en teknikk for konstruksjon av en kurve. [2] [3]

Spline er en metode for å estimere verdien av en funksjon mellom et sett av veipunkter. Dette kan brukes hvis polynomet S(x) har n antall x-verdier. Det er vanlig å sette graden til polynomet høyere enn 1 (n > 1), slik at man tilpasse den første og andre deriverte av funksjonen og dermed får vi hastighet og akselerasjon. Koblingspunkter mellom disse segmentene kalles skjøtepunkter.

Det er tre forskjellige måter å definere polynom likning av en spline kurve. I dette tilfelle brukes kubisk form som er presentert i R_2.

- **Polynomial likning** Spline funksjon kan skrives i form av standard polynom likning som vist nedenfor.

$$\mathrm{S(t)} = \begin{bmatrix} \mathrm{x(t)} \\ \mathrm{y(t)} \end{bmatrix} = \begin{bmatrix} a_x t^3 + b_x t^2 + c_x t + d_x \\ a_y t^3 + b_y t^2 + c_y t + d_y \end{bmatrix} \tag{3.1}$$

$$= \begin{bmatrix} t^3 & t^2 & t & 1 \end{bmatrix} \begin{bmatrix} a_x & a_y \\ b_x & b_y \\ c_x & c_y \\ d_x & d_y \end{bmatrix} = T \,.\, C \tag{3.2}$$

Der T representerer graden til polynom likningen, og C representerer koeffisentene i x og y koordinat system.

- **Summen av kontroll vektorer** Denne notasjonen tolker funksjonen som en lineær kombinasjon av kontroll punktene som er presentert i R_2 multiplisert med basisfunksjonen $B_i\,(t)$.

  $$\mathrm{S(t)} = \sum_{i=0}^{3} B_i\,(t)\, P_i \tag{3.3}$$

- **Matriseprodukt** Her kan Spline funksjon skrives som produkt av tre matriser.

$$\mathrm{S(t)} = T.M.G \tag{3.4}$$

T er en vektor som inneholder parameteren med graden til polynomet. C er delt opp i to matriser M som er basis matrise og G som er geometrisk matrise. Disse er definert nedenfor.

$$M = \begin{bmatrix} m_{00} & m_{01} & m_{02} & m_{03} \\ m_{10} & m_{12} & m_{12} & m_{13} \\ m_{20} & m_{13} & m_{22} & m_{23} \\ m_{30} & m_{14} & m_{32} & m_{33} \end{bmatrix},\ G = \begin{bmatrix} G_0 \\ G_1 \\ G_2 \\ G_3 \end{bmatrix} \tag{3.5}$$

G matrise inneholder kontroll vektorer mens basis matrise M ganges med $T = [\ t^3 \quad t^2 \quad 1\ ]$. Begge beskriver form av Spline funksjon. I dette tilfelle er det valgt kubisk spline der graden av polynomet er satt lik 3.

$$T.M = \begin{bmatrix} m_{00}t^3 + m_{10}t^2 + m_{20}t + m_{30} \\ m_{01}t^3 + m_{11}t^2 + mc_{21}t + m_{31} \\ m_{02}t^3 + m_{12}t^2 + mc_{22}t + m_{32} \\ m_{03}t^3 + m_{13}t^2 + mc_{23}t + m_{33} \end{bmatrix}^T \tag{3.6}$$

Det antas at tiden t ligger mellom [0,1] for hver S(t) og A^T er transponerte matrisen av A

3.2 Egenskaper

Kontinuitet:

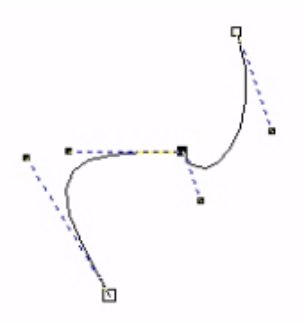
Det som er interessant med Spline metoden er at den har kontinuerlige egenskaper. Det er viktig at Spline kurven gir kontinuitet i hastighet og akselerasjon mellom to segmenter. Det antas en Kubisk Spline som er en kubisk polynom likning slik som vist nedenfor. I dette tilfelle vil banen representeres i to dimensjonalt koordinatsystemet

$$S_i(t) = a_i t^3 + b_i t^2 + c_i t + d_i, \quad t \in [0,1] \tag{3.7}$$

Dette kalles et segment $S_i(t)$ med m antall segmenter S_i, $i \in [0, m-1]$.

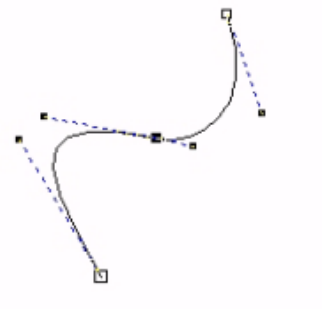
Vi skal finne ut hvilke betingelser som må være tilstede dersom to segment skal henge sammen. Her ser vi på tre grader av geometrisk kontinuitet [4][3]:

- C^0 Kontinuitet er når to kurvesegmenter har et felles punkt. Kurvene henger sammen men har en tydelig knekk. De deriverte for høyre segment og venstre segment i skjøtepunktet (figur 3-1) er forskjellige både i retning og størrelse.
  $S_i(1) = S_{i+1}(0), \ i \in [0, m-2].$ (3.8)



Figur (3-1) C^0 Kontinuitet mellom to kurvesegmenter

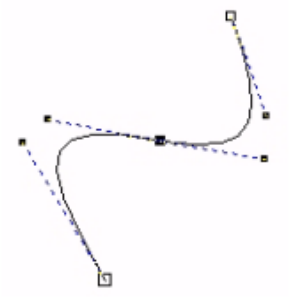
- C^1 Kontinuitet: Når to kurvesegmenter har et felles punkt og skjøten er ganske glatt. Den deriverte av høyre segment og venstre segment i skjøtepunktet har samme retning.
  $S'_i(1) = S'_{i+1}(0), \ i \in [0, m-2].$ (3.9)



Figur (3-2) C^1Kontinuitet mellom to kurvesegmenter

- C^2Kontinuitet: når to kurvesegmenter har et felles punkt, og skjøten er glatt. Den deriverte for begge segmentene har samme retning og samme størrelse.

$$S''_i(1) = S''_{i+1}(0),\ i \in [0, m-2]. \quad (3.10)$$



Figur (3-3) C^2Kontinuitet mellom to kurvesegmenter

Lokalisme:

Lokalisme er en egenskap som er viktig for å modifisere et segment i en Spline kurve som består av flere segmenter. Det vil si en Spline kurve har lokalisme hvis man har mulighet til å endre et punkt i et segment uten å påvirke resten av segmenter. Denne egenskap gir mulighet til å forandre formen lokalt uten å revurdere globale formen av kurven. Et eksempel er at hvis det oppstår hindring i et bestemt området hvor man er nødt til å justere et punkt i et segment [2].

Det vil starte med å presentere forskjellige type av spline metoder og vil fokusere på kontinuitetsegenskaper for hver metode.

3.3.1 Bezier Spline

Det antas en Kubisk Bezier kurv som inneholder fire kontroll punkter. Kurven er skrevet i geometrisk matrise G. Hvor $P_0\ og\ P_3$ passerer gjennom kurven mens $P_1\ og\ P_2$ vil estimeres.

Det starter med å definere geometrisk og basis matriser

$$G = \begin{bmatrix} P_0 \\ P_1 \\ P_2 \\ P_3 \end{bmatrix} \tag{3.11}$$

$$M = \begin{bmatrix} -1 & 3 & -3 & 1 \\ 3 & -6 & 3 & 0 \\ -3 & 3 & 0 & 0 \\ 1 & 0 & 0 & 0 \end{bmatrix} \tag{3.12}$$

Følgende likning er en Bezier kurv som er definert for fire kontroll punkter, hver punkt representert i x og y planet.

$$\mathrm{S(t)} = T.M.G \tag{3.13}$$

Da får vi følgende Bezier basis matrise:

$$B = \begin{bmatrix} B_0 \\ B_1 \\ B_2 \\ B_3 \end{bmatrix}^T = \begin{bmatrix} -t^3 + 3t^2 - 3t + 1 \\ 3t^3 - 6t^2 + 3t \\ -3t^3 + 3t^2 \\ t^3 \end{bmatrix} \tag{3.14}$$

Den er kjent også som tredje grad Bernstein polynomial likning

$$S(t) = (1-t)^3 P_0 + 3t(1-t)^2 P_1 + 3t^2(1-t)P_2 + t^3 P_3 \tag{3.15}$$

Hvor $B_i \in [0,1], i = 0 \ldots 3.$ og $\sum_{i=0}^{3} B_i = 1$

Ved å beregne C matrise fra M og G, får vi følgende resultat

$$C = M.G = \begin{bmatrix} -P_0 + 3P_1 - 3P_2 + P_3 \\ 3P_0 - 6P_1 + 3P_2 \\ -3P_0 + 3P_1 \\ P_0 \end{bmatrix} \tag{3.16}$$

Her skrives kubisk Bezier likningen som funksjon av t.

$$S(t) = (-P_0 + 3P_1 - 3P_2 + P_3)t^3 + (3P_0 - 6P_1 + 3P_2)t^2 + (-3P_0 + 3P_1)t + P_0 \tag{3.17}$$

Likningen ovenfor kan deriveres slik at vi får hastighet og akselerasjon.

$$S'(t) = (-3P_0 + 9P_1 - 9P_2 + 3P_3)t^2 + (6P_0 - 12P_1 + 6P_2)t + (-3P_0 + 3P_1) \tag{3.18}$$

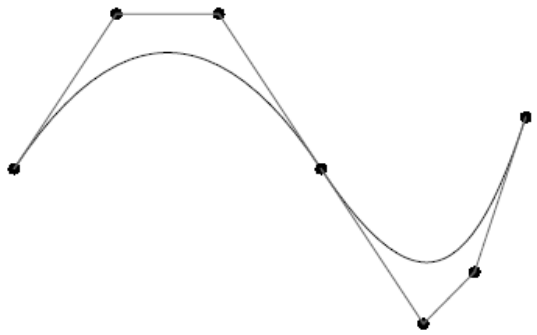
$$S''(t) = (-6P_0 + 18P_1 - 18P_2 + 6P_3)t + (6P_0 - 12P_1 + 6P_2) \tag{3.19}$$

For å finne endepunkter for hver segment settes det $S'(0)$ og $S'(1)$ slik som vist nedenfor

$$S'(0) = 3(P_1 - P_0) \quad (3.20)$$
$$S'(1) = 3(P_3 - P_2) \quad (3.21)$$

$$S''(0) = 6(P_0 - 2P_1 + P_2) \quad (3.22)$$

$$S''(1) = 6(P_1 - 2P_2 + P_3) \quad (3.23)$$



Figur (3-4) Kubikk Bezier Spline [3]

Figur (3-5) viser Kubisk Bezier Spline kurv med 2 segmenter og 7 kontroll punkter, hvor kurven passerer gjennom endepunktene for hver segment.

Egenskaper for Bezier Spline kurv:

- Kontinuitet i første deriverte
- Mangler Kontinuitet i andre deriverte
- Kurven passerer ikke gjennom første og siste punkt.

3.3.3 Kubisk Hermite spline

Kubisk Hermite Spline er en tredjegradspolynom som tar i utgangspunkt kubisk Bezier kurven, men her antas det at man kjenner kurvens endepunkter og den deriverte i endepunktene. Slik som vist i følgende matrise:

$$G = \begin{bmatrix} P_0 \\ P_3 \\ V_0 \\ V_3 \end{bmatrix} \tag{3.24}$$

G matrise uttrykker de geometriske føringene, hvor $P_0 \; og \; P_3$ antas som endepunktene, V_0 er hastighet i starten og V_3 er hastighet i slutten.

Her beregnes basis matrise som uttrykker vektfunksjonene:

$$M = \begin{bmatrix} 2 & -2 & 1 & 1 \\ -3 & 3 & -2 & -1 \\ 0 & 0 & 1 & 0 \\ 1 & 0 & 0 & 0 \end{bmatrix} \tag{3.25}$$

Med $= [\, t^3 \;\; t^2 \;\; t \;\; 1 \;\;]$ matrise får vi

$$\mathrm{S(t)} = T.M.G \tag{3.26}$$

Dette gir følgende likning for Hermite funksjon:

$$S(t) = (\, 2t^3 - 3t^2 + 1)P_0 + (-2t^3 + 3t^2)P_3 + (\, t^3 - 2t^2 + t)R_0 + (\, t^3 - \; t^2)\, R_3 \tag{3.27}$$

Da beregnes C matrise

$$C = M\,.\,G \; = \begin{bmatrix} 2P_0 - 2P_3 + R_0 + R_3 \\ -3P_0 + 3P_3 - 2R_0 - R_3 \\ R_0 \\ P_0 \end{bmatrix} \tag{3.28}$$

Dette gir:

$$S(t) = (2P_0 - 2P_3 + R_0 + R_3)t^3 + (-3P_0 + 3P_3 - 2R_0 - R_3)t^2 + R_0 t + P_0 \tag{3.29}$$

<u>Egenskaper</u>

- Kontinuitet i første deriverte.
- Mangler kontinuitet i andre deriverte.
- Kurven passerer gjennom endepunkter.

Det er ikke mulig å få kontinuitet i andre deriverte for de overnevnte spline metoder.Derfor vil det prøve å finne ut om B-spline kan løse dette problemet.

3.3.1 B-Splines

For å definere en kubisk B-spline som er en tredjegradspolynom, vil det velges fire kontroll punkter $P_0 \dots \dots P_m$ med i antall segmenter hvor $i \in [0, m-3]$. Sigmentet kan skrives på følgende likning.

$$\mathrm{S}_i(\mathrm{t}) = T.M.G \tag{3.30}$$

$$\mathrm{S}_i(\mathrm{t}) = [\, t^3 \;\; t^2 \;\; t \;\; 1 \,] \frac{1}{6} \begin{bmatrix} -1 & 3 & -3 & 1 \\ 3 & -6 & 3 & 0 \\ -3 & 0 & 3 & 0 \\ 1 & 4 & 1 & 0 \end{bmatrix} \begin{bmatrix} P_i \\ P_{i+1} \\ P_{i+2} \\ P_{i+3} \end{bmatrix} \tag{3.31}$$

Ved å beregne basis funksjonen får vi følgende matrise:

$$B = \begin{bmatrix} B_0 \\ B_1 \\ B_2 \\ B_3 \end{bmatrix}^T = \frac{1}{6} \begin{bmatrix} -t^3 + 3t^2 - 3t + 1 \\ 3t^3 - 6t^2 + 4 \\ -3t^3 + 3t^2 + 3t + 1 \\ t^3 \end{bmatrix} \tag{3.32}$$

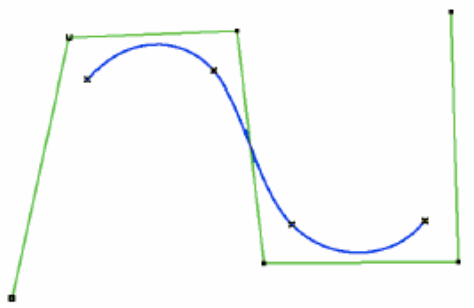
Hvor $\sum_{i=0}^{3} B_i = 1$, og $B_i \in [0,1],\ i = 0, \dots, 3,$

Her fortsetter beregningen av matrise

$$C = M\,.\,G = \begin{bmatrix} -P_i + 3P_{i+1} - 3P_{i+2} + P_{i+3} \\ 3P_i - 6P_{i+1} + 3P_{i+2} \\ -3P_i + 3P_{i+2} \\ P_i + 4P_{i+1} + P_{i+2} \end{bmatrix} \tag{3.33}$$

Da får vi følgende polynomial likning

$$S_i(t) = \frac{1}{6}(-P_i + 3P_{i+1} - 3P_{i+2} + P_{i+3})\ t^3 + \frac{1}{6}(3P_i - 6P_{i+1} + 3P_{i+2})\ t^2 + (-3P_i + 3P_{i+2})t + \frac{1}{6}(P_i + 4P_{i+1} + P_{i+2}) \tag{3.34}$$



Figur (3-5) B-Spline kurv

Egenskaper for B-spline kurve

- Kontinuitet i første deriverte
- Kontinuitet i andre deriverte
- Kurven passerer ikke gjennom kontrollpunkter

Fordeling med kubisk B-spline metoden er at man får første og andre deriverte av funksjonen, dermed får vi kontinuerlig hastighet og akselerasjon. Men vi ser at kurven passerer ikke gjennom kontroll punkter. Derfor metoden kan ikke tilfredsstille kravet for å finne tid, hastighet og akselerasjon til endepunkter for hver segment. Det finnes en måte å tvinge kuven til å treffe kontrollpunkter. B-spline kurven består av segmenter, kontrollpunkter og skjøtepunkter. Skjøtepunkter kan tvinge kurven til å passere gjennom kontrollpunkter. I tillegg spiller disse punktene stor rolle når man endrer et segment dvs. '' lokal endring '' vil det da ikke påvirker hele kurven.

Det er to type b-spline kurve, uniform og ikke-uniform basert på spredning av skjøtepunkter. Hvis avstand mellom to nabo skjøtepunkter er like, kalles det *''uniform b-spline kurve''*, mens varierende avstand mellom to nabo skjøtepunkter kalles *''ikke uniform b-spline kurve''*. [3]

3.3.4 Naturlig Kubisk Spline

Målet er å få en kuve som treffer kontroll punkter, og vil kurven oppfyller kravet for kontinuiteten i $C^0, C^1 \, og \, C^2$. [4]

Det antas en tredjegradspolynom likning $S_i(t) = a_i t^3 + b_i t^2 + c_i t + d_i$ hvor $S_i(t)$ er et segment mellom to kontrollpunkter P_i og P_{i+1}

For å få C^0 så har vi $S_i(t_i) = y_i$,og $S_i(t_{i+1}) = y_{i+1}$

$$y_i = a_i t^3 + b_i t^2 + c_i t + d_i \tag{3.35}$$

$$y_{i+1} = a_{i+1} t^3 + b_{i+1} t^2 + c_{i+1} t + d_{i+1} \tag{3.36}$$

C^0 Kontinuitet oppfyller ikke kravet for C^1 kontinuitet, dermed vil den deriverte av likningen mellom to kontrollpunkter løse problemet. Derivasjon av to nabo segmenter i ett koblingspunkt vil være like.

$$S'_i(t_{i+1}) = S'_{i+1}(t_{i+1}) \tag{3.37}$$

Dette gir følgende resultat:

$$c_i + 2b_i t_{i+1} + 3a_i t^2{}_{i+1} - c_{i+1} - 2b_{i+1} t_{i+1} - 3a_{i+1} t^2{}_{i+1} = 0 \tag{3.38}$$

Tredje krav er C^2 hvor segmentet er en kontinuerlig funksjon:

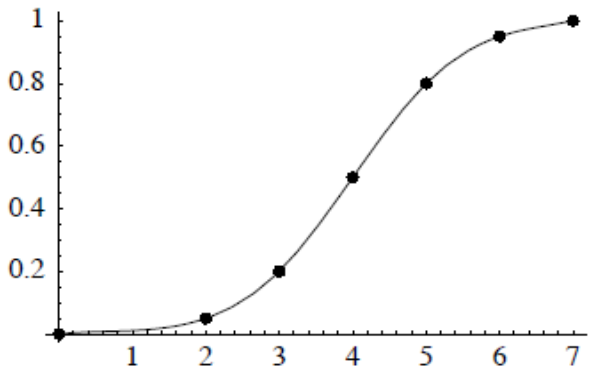
$$S''_i(t_{i+1}) = S''_{i+1}(t_{i+1}) \tag{3.39}$$

$$2b_i + 6a_i t_{i+1} - 2b_{i+1} - 6a_{i+1} t_{i+1} = 0 \tag{3.40}$$

For å fullføre kontinuitetskravet for Spline kurven, legges det til to hastighetslikninger i det endepunktene

$$S'_0(t_0) = 2a_0 t^2 + b_0 t + c_0 \tag{3.41}$$

$$S'_{n-1}(t_n) = 2a_{n-1} t^2 + b_{n-1} t + c_{n-1} \tag{3.42}$$



Figur (3-6) naturlig kubisk spline [3]

Egenskaper for naturlig kubisk spline kurve:

- Kontinuitet i første deriverte
- Kontinuitet i andre deriverte
- Kurven passerer gjennom kontrollpunkter (veipunkter)
- Mangler på lokalkontroll dvs. endring i posisjonen for et punkt påvirker hele kurven.

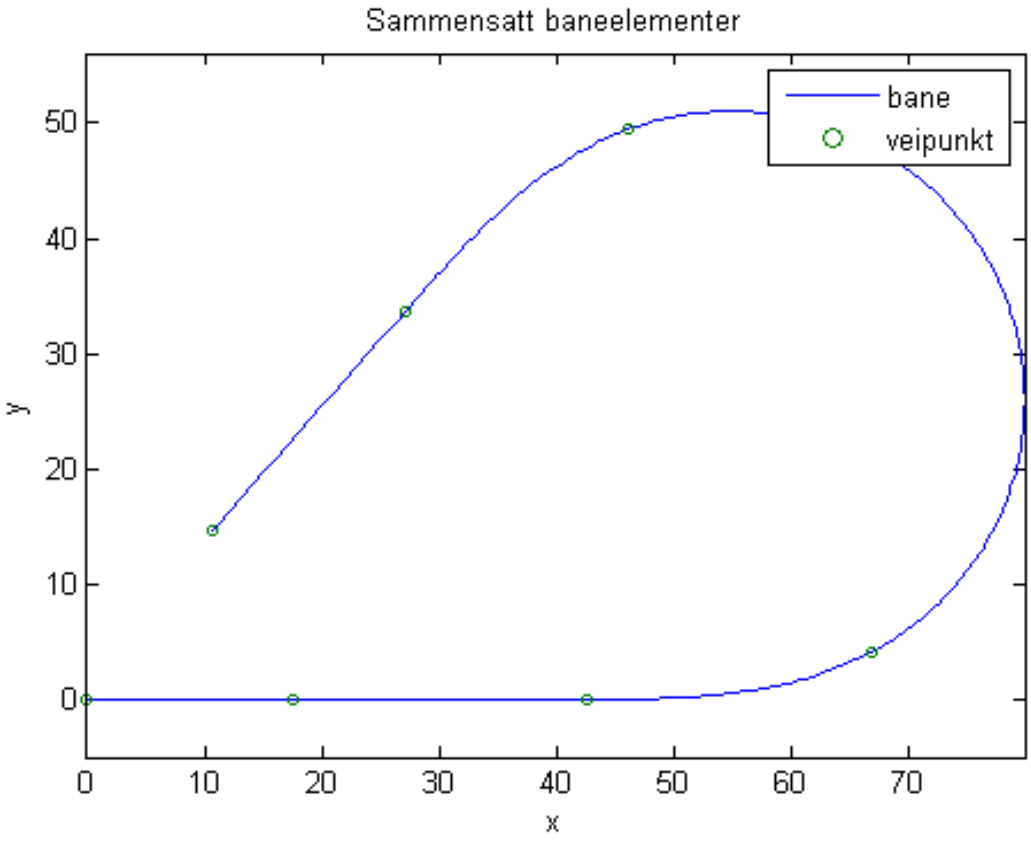
  Siden vi i dette tilfellet ikke er interessert i lokalkontroll vil det være nyttig å bruke ovennevnt algoritmen.

4 Banegenerator.

Analysering av en banegenerator bygges av tre type moduler, en for å følge rett strekning, en for å følge svingen og en for overgang mellom rettlinjen og svingen. I de ulike modulene er det gitt at forskjellige betingelser må tas hensyn til ettersom hvilken retning et legemet beveger seg.

I modulene ønskes det at posisjonen, hastighet og akselerasjon registreres med tanke på tiden, slik at legemet skifter modul til å bevege seg rett frem eller start en sving. For at fartøyet kunne garantert følge en bane som er definert i form av en kurve, så er det noen krav som vil ta hensyn til. For det meste er kontinuitets egenskaper. Med kontinuitets egenskaper menes kontinuitet i hastighet og akselerasjon.

I dette kapittel fremstilles den teoretiske tankegangen bak systemet med matematisk grunnlag for hver modul hvor riktig fremgangsmåte og utregninger har stor betydning.



Figur (4-1) Sammensatt baneelementer

Den blåe stekningen i figuren (4.1) viser banen som starter fra 0 og de skjøte punktene plottet ut som '' o''. For å realisere banen blir det utført bestemte utregninger og algoritmer bestående av vektorer og matriser. Funksjoner rettet mot dette, produserer linjer mellom to punkter. Mer beskrivelse av funksjonaliteten til hver funksjon vil bli beskrevet og fremstilt videre.

4.1 Banegenerator basert på rettlinje og sirkelbue

For å finne modulene som er nevnt ovenfor, har banen delt opp i fem baneelementer, hvor første baneelement følger rettlinje, andre tar i utgangspunkt overgang fra rettlinje til sving, tredje representerer banen i sving, fjerde for overgang fra sving til en rettlinje og siste baneelement fortsetter rettfram. Matematisk grunnlag av disse baneelementer er beskrevet i dette kapittel.

4.1.1 Rett bane:

Det antas at fartøyet starter bevegelse med å øke hastigheten fra V_0 til V_1 og akselererer samtidig. Det vil finne hastighet og posisjon ved å integrere akselerasjon likningen for første og andre gang. Dette er vist i følgende likninger:

$$\omega = \frac{2\pi}{T}\ ; A = \frac{V_1 - V_0}{T}; \underline{p}^a(T) = \begin{bmatrix} 1/2(V_1 + V_0)T \\ 0 \end{bmatrix} \tag{4.1}$$

$$\underline{p}^a(t) = \begin{bmatrix} V_0\, t + \frac{1}{2}At^2 - A/\omega^2(1 - cos\omega t) \\ 0 \end{bmatrix} \tag{4.2}$$

$$\underline{v}^a(t) = \begin{bmatrix} V_0 + A(t - \frac{1}{\omega} sin\omega t \\ 0 \end{bmatrix} \tag{4.3}$$

$$\underline{a}^a(t) = \begin{bmatrix} A(t - cos\omega t \\ 0 \end{bmatrix} \tag{4.4}$$

Deretter antas det at fartøyet forsetter med rettlinje uten akselerasjon. Da er følgende likninger for posisjon, hastighet og akselerasjon:

$$\underline{p}^a(t) = \begin{bmatrix} V_0\, t \\ 0 \end{bmatrix}; \underline{v}^a(t) = \begin{bmatrix} V_0 \\ 0 \end{bmatrix};\ \underline{a}^a(t) = \underline{0} \tag{4.5}$$

Det er laget en funksjon i MATLAB programmet som inneholder den delen av bane. Funksjonen kalles *RBA*.m.

4.1.2 Eulerspiral

Når fartøyet endrer stillingen fra rettlinje til en krum bevegelse som følger en kurve, vil sentrifugalakselerasjon endrer seg. Derfor brukes det klotoide «Eulerspiral» som en overgangskurv [5]. En viktig egenskap ved klotoiden er at den krummer lineært og proporsjonalt med kurvens lengde. Dette passer bra fordi sentrifugalkraften endrer seg på samme måte, og dermed vil klotoiden med en slik spiral fasong gir en myk overgang. I to dimensjonal kurve vil Eulerspiral beskreves med følgende parametere:

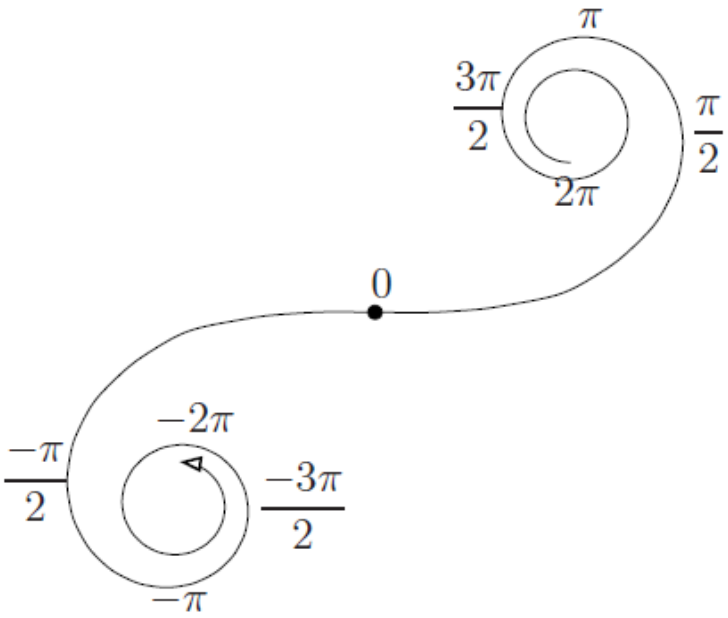
$$\begin{bmatrix} x(\theta) \\ y(\theta) \end{bmatrix} = \begin{bmatrix} cos(\theta) \\ sin(\theta) \end{bmatrix} \tag{4.6}$$

Hvor $cos(\theta)$ og $sin(\theta)$ er definert slik

$$cos(\theta) = \frac{1}{\sqrt{2\pi}} \int_0^{\theta} \frac{\cos(\tau)}{\sqrt{\tau}} d\tau \tag{4.7}$$

$$sin(\theta) = \frac{1}{\sqrt{2\pi}} \int_0^{\theta} \frac{\sin(\tau)}{\sqrt{\tau}} d\tau \tag{4.8}$$

Hvor θ er vinkelen mellom tangenten og positiv x-aksen.



Figur (4-2) Eulerspiral med θverdier

Figur (4-2) viser Eulerspiral hvor positiv verdi av t og θ ligger på høyre side og en negativ verdi på venstre side. Slik spiral fasongen kan benyttes her ved overgang fra rettlinje til sirkelbue.

Eulerspiral kan utledes på to måter som begge gir samme resultat.

- *Utledning 1*

Den første utledningen tar i utgangspunkt sentripetal akselerasjon som er normalt på hastighetsvektor og øker lineært. Det vil finne posisjon, hastighet og akselerasjon

$$\dot{\underline{p}}^a = \underline{v}^a \tag{4.9}$$

$$\dot{\underline{v}}^a = \underline{a}^a \tag{4.10}$$

Der $\underline{a}^a = \begin{bmatrix} 0 \\ At/T \end{bmatrix}$ Som gir akselerasjon langs $\vec{b}_2$

$$f^b: \vec{b}_1 = \frac{\underline{v}^a}{\|\underline{v}^a\|} \tag{4.11}$$

$$\vec{b}_2 = \vec{b}_3 \, X \, \vec{b}_1 \tag{4.12}$$

$$R_b^a = [\underline{b}_1^a \, ; \, \underline{b}_2^a ; \, \underline{b}_3^a] \tag{4.13}$$

$$\underline{v}^a = R_b^a \, \underline{v}^{ab} = [\underline{v}_1^{ab} \underline{b}_1^a \, ; \, \underline{v}_2^{ab} \underline{b}_2^a \, ; \, \underline{0} \,] \tag{4.14}$$

$$\underline{a}^a = R_b^a \, \underline{a}^{aab} = R_b^a \begin{bmatrix} 0 \\ At/T \\ 0 \end{bmatrix} = \underline{b}_2^a \, At/T \tag{4.15}$$

$$\underline{b}_2^a = \underline{b}_3^a \, X \, \underline{b}_1^a \tag{4.16}$$

Ved å innføre en skjevsymmetrisk form av vektoren $\vec{b}_3$ kan matrisen skrives som $\boldsymbol{S}(\underline{b}_3^a)$. Denne formen kan også skrives som en "$\vec{b}_3 \, X$ " operator, og i en ortonormal basis, $\{\vec{a}_i\}$ er den gitt ved:

$$\boldsymbol{S}(\underline{b}_3^a) \, \underline{b}_1^a = \begin{bmatrix} 0 & -1 & 0 \\ 1 & 0 & 0 \\ 0 & 0 & 0 \end{bmatrix} . \underline{b}_1^a = [-\underline{b}_{21}^a \, ; \, \underline{b}_{11}^a ; \, 0] \tag{4.17}$$

$$\underline{b}_3^a = [\, 0 \, ; \, 0 \, ; \, 1 \,] \tag{4.18}$$

$$\underline{b}_1^a = [\underline{b}_{11}^a \, ; \, \underline{b}_{21}^a ; \, \underline{b}_{31}^a] \tag{4.19}$$

$$\underline{b}_1^a = \frac{\underline{v}^a}{\|\underline{v}^a\|} \tag{4.20}$$

$$\underline{b}_1^a = \frac{1}{v} . \underline{v}^a = \frac{1}{v} [\underline{v}_1^a \, ; \, \underline{v}_2^a ; 0] \tag{4.21}$$

$$\underline{b}_2^a = \frac{1}{v} [-\underline{v}_2^a \, ; \, \underline{v}_1^a] \tag{4.22}$$

$$\dot{\underline{\boldsymbol{v}}}^{\boldsymbol{a}} = \underline{b}_2^a . At = \frac{At}{VT} \begin{bmatrix} -\underline{v}_2^a \\ \underline{v}_1^a \end{bmatrix} = \frac{At}{VT} \begin{bmatrix} 0 & -1 \\ 1 & 0 \end{bmatrix} . \underline{v}^a \tag{4.23}$$

$$\dot{\underline{\boldsymbol{p}}}^{\boldsymbol{a}} = \underline{v}^a \tag{4.24}$$

$$\dot{\underline{\boldsymbol{v}}}^{\boldsymbol{a}} = \boldsymbol{F}(\boldsymbol{t}) \, \underline{v}^a \quad \text{hvor} \quad (\boldsymbol{t}) = \frac{At}{VT} \begin{bmatrix} 0 & -1 \\ 1 & 0 \end{bmatrix} \tag{4.25}$$

$$\underline{v}^a \, (\boldsymbol{t}) = \boldsymbol{e}^{\frac{At^2}{2VT} \begin{bmatrix} 0 & -1 \\ 1 & 0 \end{bmatrix}} \, \underline{v}^a(0) \tag{4.26}$$

Utledning 1 gir oss følgende resultat:

$$\underline{p}_0^a = \begin{bmatrix}0\\0\end{bmatrix} ;\ \underline{v}^a\ (\mathbf{0}) = \begin{bmatrix}V\\0\end{bmatrix} ; \theta(T) = arctan\frac{v_2^a(\boldsymbol{T})}{v_1^a(\boldsymbol{T})} \tag{4.27}$$

$$\underline{p}^a(t) = \underline{p}_0^a + (\int_0^t \underline{v}^a(s)\, ds)\underline{v}_0^a \tag{4.28}$$

$$\underline{v}^a(t) = \boldsymbol{e}^{\frac{At^2}{2VT}\begin{bmatrix}0 & -1\\1 & 0\end{bmatrix}}\ \underline{v}^a(0) \tag{4.29}$$

$$\underline{a}^a(t) = \frac{At}{VT}\begin{bmatrix}0 & -1\\1 & 0\end{bmatrix} \boldsymbol{e}^{\frac{At^2}{2VT}\begin{bmatrix}0 & -1\\1 & 0\end{bmatrix}}\ \underline{v}^a(0) \tag{4.30}$$

For å få posisjon integreres det hastighetslikning som er en eksponential funksjon. Integrasjon av en sånn funksjon vil ende med en numerisk feil. For å unngå den numerisk feilen beregnes Eulerspiral på en annen måte.

- *Utledning 2*

Utledning 2 tar hensyn til vinkelhastighet i banen. Siden banen konsentrerer kun om x og y planet brukes en 2-dimensjonal rotasjonsmatrise på formen:

$$R(\theta_3) = \begin{bmatrix}\cos\theta_3 & -\sin\theta_3\\ \sin\theta_3 & \cos\theta_3\end{bmatrix} \tag{4.31}$$

$$\underline{v}^a(t) = R_b^a\ \underline{v}^{ab} \tag{4.32}$$

Hvis det antas at akselerasjonen a(t) øker fra 0 til A i løpet av T vil vi få

$$a(t) = \frac{At}{T} = \dot{\theta}V\ ;\ \dot{\theta}V = At/VT \tag{4.34}$$

$$\theta(t) = \frac{At^2}{2VT} \tag{4.35}$$

Dette gir følgende likninger for posisjon, hastighet og akselerasjon når det antas hastighet er konstant og akselerasjon er normalt på hastighetsvektor:

$$\underline{v}^a(t) = \begin{bmatrix}\cos\theta_3 & -\sin\theta_3\\ \sin\theta_3 & \cos\theta_3\end{bmatrix}\begin{bmatrix}V\\0\end{bmatrix} = V\begin{bmatrix}cos\theta\\ sin\theta\end{bmatrix} \tag{4.36}$$

$$\underline{a}^a(t) = \dot{\theta}V\begin{bmatrix}-sin\theta\\ cos\theta\end{bmatrix} \tag{4.37}$$

$$\underline{p}^a(t) = \int_0^t V\begin{bmatrix}cos\theta\\ sin\theta\end{bmatrix} ds \tag{4.38}$$

$$\underline{p}^a(t) = \int_0^t V \begin{bmatrix} cos\frac{A}{2VT}s^2 \\ sin\frac{A}{2VT}s^2 \end{bmatrix} ds \quad (4.39)$$

Vi vet at fresnel integral kan utledes fra Eulerspiral og gir følgende likninger:

$$cos(t) = \int_0^t \cos\frac{\pi}{2}\tau^2 d\tau \quad (4.40)$$

$$sin(t) = \int_0^t \sin\frac{\pi}{2}\tau^2 d\tau \quad (4.41)$$

Det brukes fresnel integral likninger til å finne posisjon. Funksjonen er definert i MatLab som fresnels for sinus funksjon og fresnelc for cosinus funksjon

Utledning 2 gir oss følgende likninger som kan implementeres i MatLab programmet. Da antas det at akselerasjon økes lineært fra 0 til A i løpet av T sekunder.

$$\underline{p}_0^a = \begin{bmatrix} 0 \\ 0 \end{bmatrix};\ \underline{v}^a\ (\mathbf{0}) = \begin{bmatrix} V \\ 0 \end{bmatrix} \quad (4.42)$$

$$\theta(t) = At^2/2VT;\ \theta(T) = AT/2V \quad (4.43)$$

$$\Omega = \sqrt{\frac{A}{\pi VT}} \quad (4.44)$$

$$\underline{p}^a(t) = \frac{V}{\Omega}\begin{bmatrix} C(\Omega t) \\ S(\Omega t) \end{bmatrix};\ \underline{v}^a(t) = V\begin{bmatrix} cos\theta(t) \\ sin\theta(t) \end{bmatrix};\ \underline{a}^a(t) = At/T\begin{bmatrix} -sin\theta(t) \\ cos\theta(t) \end{bmatrix} \quad (4.45)$$

Det er lagt inn en egen funksjon for disse likningene i MatLab programmet. Funksjonen kalles ES.m som er beskrevet i avsnitt 4.3.3

4.1.3 Sirkelbane

Det antas at banen svinger videre til venstre da skal det brukes følgende likninger for en sirkelbevegelse:

$\omega = \frac{2\pi}{T_\omega} = \frac{A}{V} = \frac{V}{R}$ hvor T_ω er tiden det tar for 360 –graders sving (4.46)

$$\theta(t) = \omega t \quad (4.47)$$

$$\underline{p}^a(t) = R\begin{bmatrix} sin\theta \\ 1 - cos\theta \end{bmatrix};\ \underline{v}^a(t) = V\begin{bmatrix} cos\theta \\ sin\theta \end{bmatrix};\ \underline{a}^a(t) = A\begin{bmatrix} -sin\theta \\ cos\theta \end{bmatrix} \quad (4.48)$$

4.1.4 Omvendt Eulerspiral

I baneelement 4 antas det at banen endrer stillingen fra sirkelbevegelse til rettstrekning, da brukes Eulerspiral igjen her, slik at vi får en jevn sving. Dette baneelement kalles omvendt Eulerspiral som er lagt inn som en egen funksjon i MATLAB programmet OES.m. I dette tilfelle skal det brukes Eulerspiral likningene som har funnet i avsnitt 4.1.2 Utledning 2. I dette tilfelle antas det at akselerasjon avtar lineært fra A til 0. Da settes tiden $t = T - t$

$$a(t) = \frac{A(T-t)}{T} = \dot{\theta}V \; ; \; \dot{\theta}(t) = A/VT(T-t) \tag{4.49}$$

$$\theta(t) = \int_0^t \frac{A}{VT(T-\tau)} d\tau = \frac{A}{VT}\left(\int_0^t T \, d\tau - \int_0^t \tau \, d\tau\right) \tag{4.50}$$

$$\boldsymbol{\theta(t) = A/V(t - t^2/2T)} \tag{4.51}$$

$$\boldsymbol{\theta(T) = AT/2V} \tag{4.52}$$

Der hastigheten og akselerasjon er

$$\underline{\boldsymbol{v}}^{\boldsymbol{a}}(\boldsymbol{t}) = \boldsymbol{V}\begin{bmatrix}\boldsymbol{cos\theta(t)}\\ \boldsymbol{sin\theta(t)}\end{bmatrix}; \tag{4.53}$$

$$\underline{\boldsymbol{a}}^{\boldsymbol{a}}(\boldsymbol{t}) = \boldsymbol{A(T-t)/T}\begin{bmatrix}\boldsymbol{-sin\theta(t)}\\ \boldsymbol{cos\theta(t)}\end{bmatrix} \tag{4.54}$$

For posisjon vil det brukes likningene fra Eulerspiral, men i dette tilfelle brukes det ramme f^h istedenfor rammef^a.

$$\Omega = \sqrt{\frac{A}{\pi VT}} \tag{4.55}$$

$$\underline{p}^h(t) = \frac{V}{\Omega}\begin{bmatrix}cos(\Omega t)\\ sin(\Omega t)\end{bmatrix}; \tag{4.56}$$

Det antas at det kan beregnes $\underline{p}^h(t)$ i den nye ramma f^a

$$\underline{p}^h(t) = \underline{p}^h + R_h^a \underline{p}^a(t) = \underline{p}^h(T) + R_h^a \underline{p}^a(t) \tag{4.57}$$

$$\underline{p}^a(t) = R_h^a(\underline{p}^h(t) - \underline{p}^h(T)) \tag{4.58}$$

For å få en bane som starter i origo med retning $\check{a}_1$ må vi erstatte t med (T-t) og endre fortegnet for $\check{a}_1$ –komponenten. Dette medfører

$$\underline{p}^a(t) = \begin{bmatrix}-1 & 0\\ 0 & 1\end{bmatrix} R_h^a(\underline{p}^h(T-t) - \underline{p}^h(T)) \tag{4.59}$$

$$\underline{p}^a(t) = \begin{bmatrix}-1 & 0\\ 0 & 1\end{bmatrix} R_3\left(-\frac{AT}{2V}\right)\frac{V}{\Omega}\begin{bmatrix}C(\Omega(T-t) - C(\Omega T)\\ S(\Omega(T-t)) - S(\Omega T)\end{bmatrix}; \tag{4.60}$$

$$\underline{p}^{a}(t) = \frac{V}{\Omega}\begin{bmatrix}-1 & 0\\ 0 & 1\end{bmatrix}\begin{bmatrix}cos\frac{AT}{2V} & sin\frac{AT}{2V}\\ -sin\frac{AT}{2V} & cos\frac{AT}{2V}\end{bmatrix} \tag{4.61}$$

Posisjonslikningen for omvendt Eulerspiral

$$\underline{\boldsymbol{p}}^{\boldsymbol{a}}(\boldsymbol{t}) = \frac{\boldsymbol{V}}{\Omega}\begin{bmatrix}-\boldsymbol{cos}\frac{\boldsymbol{AT}}{\boldsymbol{2V}} & -\boldsymbol{sin}\frac{\boldsymbol{AT}}{\boldsymbol{2V}}\\ -\boldsymbol{sin}\frac{\boldsymbol{AT}}{\boldsymbol{2V}} & \boldsymbol{cos}\frac{\boldsymbol{AT}}{\boldsymbol{2V}}\end{bmatrix}\begin{bmatrix}\boldsymbol{C}(\Omega(\boldsymbol{T}-\boldsymbol{t})-\boldsymbol{C}(\Omega\boldsymbol{T})\\ \boldsymbol{S}(\Omega(\boldsymbol{T}-\boldsymbol{t}))-\boldsymbol{S}(\Omega\boldsymbol{T})\end{bmatrix}; \tag{4.62}$$

4.2 Banegenerator basert på Spline metoden

Siden Spline er en delvis parametrisk polynomfunksjon, undersøkes det først om polynomligningen kan være en tilfredsstillende algoritme til oppbygging av banegenerator.

4.2.1 Polynombane

Polynom funksjon har mange attraktive matematiske egenskaper. Det som er interessant er kontinuerlige egenskaper. Det vil finne ut om funksjonen kan gi oss kontinuerlig hastighet og akselerasjon.

Vi starter med å generaliserer utledningen for femtegradspolynom. Det antas at start og slutt verdiene for posisjon, hastighet og akselerasjon er kjent. Dette gir oss to ligningssett, hver med seks ligninger.

$$\underline{P}^a(t) = \begin{bmatrix} {p^a}_a(t) \\ {p^a}_b(t) \end{bmatrix} = \begin{bmatrix} \sum_{j=0}^{5} a_i^j \, t^j \\ \sum_{j=0}^{5} b_i^j \, t^j \end{bmatrix} \; hvor \; t\epsilon[t_i \, , t_{i+1} \,] og \; i = 1{,}2 \, , \ldots . N \tag{4.63}$$

Der vi spesifiserer disse initialverdier

$$\underline{P}^a(t_1), \underline{P}^a(t_2), \underline{v}^a(t_1), \underline{v}^a(t_2), \underline{a}^a(t_1) \; og \; \underline{a}^a(t_2) \tag{4.64}$$

Dersom vi definerer

$$\underline{a_i} = \begin{bmatrix} {a^5}_i \\ {a^4}_i \\ {a^3}_i \\ {a^2}_i \\ {a^1}_i \\ {a^0}_i \end{bmatrix} \; og \; \underline{b_i} = \begin{bmatrix} {b^5}_i \\ {b^4}_i \\ {b^3}_i \\ {b^2}_i \\ {b^1}_i \\ {b^0}_i \end{bmatrix} \tag{4.65}$$

Der $\underline{a_i}$ og $\underline{b_i}$ er koeffisienter i to dimensjonalt koordinatsystem

Det vil beskrive femte gradsparameter vektor for posisjon, hastighet og akselerasjon på følgende måte:

$$P(t) = [\, t^5 \;\; t^4 \, t^3 \;\; t^2 \;\; t \;\; 1 \;\;] \quad hvor \; P_i = P \, (t_i) \tag{4.66}$$

$$V(t) = [\; 5t^4 4t^3 \;\; 3t^2 \;\; 2\, t \;\; 1 \;\; 0 \;\;] \quad hvor \; V_i = V \, (t_i) \tag{4.67}$$

$$A(t) = [20t^3 \;\; 12t^2 \;\; 6t \;\; 2 \;\; 0 \;\; 0] \quad hvor \;\; A_i = A \, (t_i) \tag{4.68}$$

Vi skriver verdiene i et lineart system

$$\boldsymbol{F}\underline{\boldsymbol{a}} = \underline{\boldsymbol{h}}_{\boldsymbol{1}}, \boldsymbol{F}\underline{\boldsymbol{b}} = \underline{\boldsymbol{h}}_{\boldsymbol{2}} \tag{4.69}$$

Hvor F representerer parameter vektor for posisjon, hastighet og akselerasjon.
$\underline{a_i}$ representerer koeffisientene.
$\underline{h}_{1,2}$ representerer initialverdier

Da får vi

$$\begin{bmatrix} P_1 \\ P_2 \\ V_1 \\ V_2 \\ A_1 \\ A_2 \end{bmatrix} [\underline{a}] = \begin{bmatrix} p_1^a(t_1) \\ p_1^a(t_2) \\ v_1^a(t_1) \\ v_1^a(t_2) \\ a_1^a(t_1) \\ a_1^a(t_1) \end{bmatrix}; \begin{bmatrix} P_1 \\ P_2 \\ V_1 \\ V_2 \\ A_1 \\ A_2 \end{bmatrix} [\underline{b}] = \begin{bmatrix} p_2^a(t_1) \\ p_2^a(t_2) \\ v_2^a(t_1) \\ v_2^a(t_2) \\ a_2^a(t_1) \\ a_2^a(t_1) \end{bmatrix} \tag{4.70}$$

$$\begin{bmatrix} P_1 & \underline{0}^T \\ \underline{0}^T & P_1 \\ P_2 & \underline{0}^T \\ \underline{0}^T & P_2 \\ V_1 & \underline{0}^T \\ \underline{0}^T & V_1 \\ V_2 & \underline{0}^T \\ \underline{0}^T & V_2 \\ A_1 & \underline{0}^T \\ \underline{0}^T & A_1 \\ A_2 & \underline{0}^T \\ \underline{0}^T & A_2 \end{bmatrix} \begin{bmatrix} \underline{a_1} \\ \underline{b_1} \end{bmatrix} = \begin{bmatrix} \underline{p}^a(t_1) \\ \underline{p}^a(t_2) \\ \underline{v}^a(t_1) \\ \underline{v}^a(t_2) \\ \underline{a}^a(t_1) \\ \underline{a}^a(t_2) \end{bmatrix} \tag{4.71}$$

4.2.2 Splinebane

I dette avsnittet vil det lages en bane som baseres på *Naturlig Spline* metoden. Det antas at det er N baneelementer som er koblet sammen med N+1 punkter i planet. Disse punktene er nummerert fra 1 til N+1. $(WP_i \ldots . WP_{i+1})$ og er representert i R_2 slik at a Representerer x-aksen og b_i representerer y-aksen.

- Naturlig 3.grads spline

De betingelsene vi satt opp i avsnitt 4.2.1 Polynombane, med å spesifisere posisjon, hastighet og akselerasjon er utgangspunktet for en navngitt type kurve ''Naturlig 3. grads spline kurve''.

Vi starter med å undersøke om Naturlig kubisk spline kurve gir oss en tilfredsstillende løsning. Det antas at $P_i(t)$ er en vektor som representerer graden til polynomet for posisjonen mellom punktet i og punktet i+1.

Da får vi

$$\underline{P}^a(t) = \begin{bmatrix} p^a{}_a(t) \\ p^a{}_a(t) \end{bmatrix} = \begin{bmatrix} \sum_{k=0}^{3} a_i^k \, t^k \\ \sum_{k=0}^{3} b_i^k \, t^k \end{bmatrix} \; hvor \; t\epsilon[t_i \, , t_{i+1} \,] og \; i = 1{,}2 \, , \ldots . N \tag{4.72}$$

$$\underline{a}_i = \begin{bmatrix} a^3{}_i \\ a^2{}_i \\ a^1{}_i \\ a^0{}_i \end{bmatrix} , \underline{b}_i = \begin{bmatrix} b^3{}_i \\ b^2{}_i \\ b^1{}_i \\ b^0{}_i \end{bmatrix} \tag{4.73}$$

Hvor $\underline{a}_i$ koeffisient matrise i x-aksen og $\underline{b}_i$ er koeffisient matrise i y –aksen.

$$P_i(t) = \begin{bmatrix} t_i^3 & t_i^2 & t_i & 1 \end{bmatrix} hvor \; P_i = P \, (t_i) \tag{4.74}$$

Ved å derivere parameters vektor for posisjon, får vi parameters vektor for hastighetsvektor

$$V(t) = \begin{bmatrix} 3t_i^2 & 2t_{\,i} & 1 & 0 \end{bmatrix} \; hvor \; V_i = V \, (t_i) \tag{4.75}$$

Deretter finner vi det siste parameter for akselerasjon

$$A(t) = [\; 6t_i \;\; 2 \;\; 0 \;\; 0] \;\; hvor A_i = A \, (t_i) \tag{4.76}$$

$$\underline{P_i}^n(t) = \begin{bmatrix} P(t) & \underline{0}^T \\ \underline{0}^T & P(t) \end{bmatrix} \cdot \begin{bmatrix} \underline{a_i} \\ \underline{b_i} \end{bmatrix} hvor \; t\epsilon[t_i \, , t_{i+1} \,] og \; i = 1{,}2 \, , \ldots . N \tag{4.77}$$

Vi antar en kurve som er sammensatt av 3 kurvesegmenter (baneelementer), Vi vil kikke litt på hvordan vi kan skjøte baneelementer og oppnå ulik grad av glatthet. Vi vil konstruerer oss om hvilke betingelser som må være tilstede dersom to baneelementer skal henge sammen på en '' glatt'' måte. Vi ser på tre grader av kontinuitet for å oppfylle kravet til våre bane.

- C^0 Kontinuitet: For hver baneelement har vi to vektorlikninger siden $\underline{P_i}^n(t_i) = \underline{P}^n(t_i)$ og $\underline{P_i}^n(t_{i+1}) = \underline{P}^n(t_{i+1})$ er gitt. Dette gir 4N likninger.

$$\begin{bmatrix} P_i & \underline{0}^T \\ \underline{0}^T & P_i \\ P_{i+1} & \underline{0}^T \\ \underline{0}^T & P_{i+1} \end{bmatrix} \cdot \begin{bmatrix} \underline{a_i} \\ \underline{b_i} \end{bmatrix} = \begin{bmatrix} \underline{P}^n(t_i) \\ \underline{P}^n(t_{i+1}) \end{bmatrix} \quad , = 1, \dots, N \tag{4.78}$$

- C^1 Kontinuitet: Hastighet ved overgang fra et baneelement til neste skal være kontinuerlig dvs. $\underline{v_i}^n(t_{i+1}) = \underline{v_{i+1}}^n(t_{i+1})$. Dette gir 2(N-1) likninger.

$$\begin{bmatrix} V_{i+1} & \underline{0}^T & -V_{i+1} & \underline{0}^T \\ \underline{0}^T & V_{i+1} & \underline{0}^T & -V_{i+1} \end{bmatrix} \cdot \begin{bmatrix} \underline{a_i} \\ \underline{b_i} \\ \underline{a_{i+1}} \\ \underline{b_{i+1}} \end{bmatrix} = \underline{0}\,;\ i = 1, \dots, N-1 \tag{4.79}$$

- C^2 Kontinuitet: akselerasjon ved overgang fra et baneelement til neste skal være kontinuerlig dvs. $\underline{a_i}^n(t_{i+1}) = \underline{a_{i+1}}^n(t_{i+1})$. Dette gir 2(N-1) likninger.

$$\begin{bmatrix} A_{i+1} & \underline{0}^T & -A_{i+1} & \underline{0}^T \\ \underline{0}^T & A_{i+1} & \underline{0}^T & -A_{i+1} \end{bmatrix} \cdot \begin{bmatrix} \underline{a_i} \\ \underline{b_i} \\ \underline{a_{i+1}} \\ \underline{b_{i+1}} \end{bmatrix} = \underline{0}\,;\ \ i = 1, \dots, N-1 \tag{4.80}$$

- Tilleggskrav: Det trenger fire likninger for å definere hastighet og akselerasjon i start og slutt punkter $\underline{v_1}^n(t_1)$, $\underline{v_N}^n(t_{N+1}), \underline{a_1}^n(t_1)$ og $\underline{a_N}^n(t_{N+1})$

$$\begin{bmatrix} V_1 & \underline{0}^T \\ \underline{0}^T & V_1 \\ A_1 & \underline{0}^T \\ \underline{0}^T & A_1 \end{bmatrix} \cdot \begin{bmatrix} \underline{a_1} \\ \underline{b_1} \end{bmatrix} = \begin{bmatrix} \underline{v}^n(t_1) \\ \underline{a}^n(t_1) \end{bmatrix} = \begin{bmatrix} v_1{}^n(t_1) \\ v_2{}^n(t_1) \\ a_1{}^n(t_1) \\ a_2{}^n(t_1) \end{bmatrix} \tag{4.81}$$

Får å finne koeffisientene $\underline{a_1}$ og $\underline{b_1}$ settes posisjonsvektor, hastighetsvektor og akselerasjonsvektor sammen slik at vi får

$$\boldsymbol{F}\underline{\boldsymbol{a}} = \underline{\boldsymbol{h_1}} \tag{4.82}$$

$$\begin{bmatrix} V_1 & 0 & 0 \\ A_1 & 0 & 0 \\ P_1 & 0 & 0 \\ P_2 & 0 & 0 \\ 0 & P_2 & 0 \\ 0 & P_3 & 0 \\ 0 & 0 & P_3 \\ 0 & 0 & P_4 \\ V_2 & -V_2 & 0 \\ 0 & V_3 & -V_3 \\ A_2 & -A_2 & 0 \\ 0 & A_3 & -A_3 \end{bmatrix} \cdot \begin{bmatrix} \underline{a}_1 \\ \underline{a}_2 \\ \underline{a}_3 \end{bmatrix} = \begin{bmatrix} {v_1}^n(t_1) \\ {a_1}^n(t_1) \\ {p_1}^n(t_1) \\ {p_1}^n(t_2) \\ {p_1}^n(t_2) \\ {p_1}^n(t_3) \\ {p_1}^n(t_3) \\ {p_1}^n(t_4) \\ 0 \\ 0 \\ 0 \\ 0 \end{bmatrix} \tag{4.83}$$

Det kan skrives på følgende form

$$\underline{\boldsymbol{a}} = \boldsymbol{F}^{-1}\underline{\boldsymbol{h}}_{\boldsymbol{1}} \tag{4.84}$$

Når det gjelder y-retning så har vi disse føringene

$$\begin{bmatrix} V_1 & 0 & 0 \\ A_1 & 0 & 0 \\ P_1 & 0 & 0 \\ P_2 & 0 & 0 \\ 0 & P_2 & 0 \\ 0 & P_3 & 0 \\ 0 & 0 & P_3 \\ 0 & 0 & P_4 \\ V_2 & -V_2 & 0 \\ 0 & V_3 & -V_3 \\ A_2 & -A_2 & 0 \\ 0 & A_3 & -A_3 \end{bmatrix} \cdot \begin{bmatrix} \underline{b}_1 \\ \underline{b}_2 \\ \underline{b}_3 \end{bmatrix} = \begin{bmatrix} {v_1}^n(t_1) \\ {a_1}^n(t_1) \\ {p_1}^n(t_1) \\ {p_1}^n(t_2) \\ {p_1}^n(t_2) \\ {p_1}^n(t_3) \\ {p_1}^n(t_3) \\ {p_1}^n(t_4) \\ 0 \\ 0 \\ 0 \\ 0 \end{bmatrix} \tag{4.85}$$

Som kan skrives slik

$$\underline{\boldsymbol{b}} = \boldsymbol{F}^{-1}\underline{\boldsymbol{h}}_{\boldsymbol{2}} \tag{4.86}$$

Etter at koeffisientene er beregnet, kan vi finne posisjon, hastighet og akselerasjon i navigasjonsramma $\boldsymbol{f^n}$.

- Naturlig 4. grads spline

Vi er ikke fornøyd med å bare ha tredje grads spline bane. Vi må kunne håndtere svingen mer avansert derfor vil det utlede ligningen for å tilpasse 4.gradskurven med naturlig spline algoritme.

Da starter vi slik

$$\underline{P}^a(t) = \begin{bmatrix} p^a{}_a(t) \\ p^a{}_b(t) \end{bmatrix} = \begin{bmatrix} \sum_{k=0}^{4} a_i^k\, t^k \\ \sum_{k=0}^{4} b_i^k\, t^k \end{bmatrix} \; hvor\; t\epsilon[t_i\,, t_{i+1}\,] og\; i = 1{,}2\,, \ldots . N \tag{4.87}$$

Koeffisientene vil skrives i følgende vektor

$$\underline{a_i} = \begin{bmatrix} a^4{}_i \\ a^3{}_i \\ a^2{}_i \\ a^1{}_i \\ a^0{}_i \end{bmatrix}, \underline{b_i} = \begin{bmatrix} b^4{}_i \\ b^3{}_i \\ b^2{}_i \\ b^1{}_i \\ b^0{}_i \end{bmatrix} \tag{4.88}$$

Hvor $\underline{a_i}$ er koeffisienten i x-aksen og $\underline{b_i}$ er koeffisienten i y –aksen.

Det kan vi fortsetter med å utlede parameter vektor for posisjon

$$P_i(t) = \begin{bmatrix} t_i^4 & t_i^3 & t_i^2 & t_i & 1 \end{bmatrix} hvor\; P_i = P\,(t_i) \tag{4.89}$$

Ved derivasjon får vi

$$V(t) = \begin{bmatrix} 4t_i^3 & 3t_i^2 & 2t_{\,i} & 1 & 0 \end{bmatrix}\; hvor\; V_i = V\,(t_i) \tag{4.90}$$

Deretter finner vi parametervektor for akselerasjon

$$A(t) = \begin{bmatrix} 12t_i^2 & 6t_i & 2 & 0 & 0 \end{bmatrix}\; hvor A_i = A\,(t_i) \tag{4.91}$$

I $\boldsymbol{f^n}$ får vi

$$\underline{P_i}^n(t) = \begin{bmatrix} P(t) & \underline{0}^T \\ \underline{0}^T & P(t) \end{bmatrix} \cdot \begin{bmatrix} \underline{a_i} \\ \underline{b_i} \end{bmatrix} hvor\; t\epsilon[t_i\,, t_{i+1}\,] og\; i = 1{,}2\,, \ldots . N \tag{4.92}$$

Vi vet at $\underline{a_i}$ har 5 antall konstanter i x-aksen og $\underline{b_i}$ har 5 antall konstanter i y-aksen. Da betyr at 10 N skal finnes ut

For å få

- C^0 Kontinuitet: For hver baneelement har vi to vektorlikninger siden $\underline{P_{ai}}^n(t_i) = \underline{p_1}^n(t_i)$ og $\underline{P_{ai}}^n(t_{i+1}) = \underline{p_1}^n(t_{i+1})$ er gitt. Dette gir 4N likninger.

$$\begin{bmatrix} P_i & \underline{0}^T \\ \underline{0}^T & P_i \\ P_{i+1} & \underline{0}^T \\ \underline{0}^T & P_{i+1} \end{bmatrix} \cdot \begin{bmatrix} \underline{a_i} \\ \underline{b_i} \end{bmatrix} = \begin{bmatrix} \underline{p_1}^n(t_i) \\ \underline{p_1}^n(t_{i+1}) \end{bmatrix} \quad , = 1, \ldots, N \tag{4.93}$$

- C^1 Kontinuitet: Hastighet ved overgang fra et baneelement til neste skal være kontinuerlig dvs. $\underline{v_{ai}}^n(t_i) = \underline{v_1}^n(t_i)$ og $\underline{v_{ai}}^n(t_{i+1}) = \underline{v_1}^n(t_{i+1})$. Dette gir 2N likninger i hver akse.

$$\begin{bmatrix} V_{i+1} & \underline{0}^T & -V_{i+1} & \underline{0}^T \\ \underline{0}^T & V_{i+1} & \underline{0}^T & -V_{i+1} \end{bmatrix} \cdot \begin{bmatrix} \underline{a}_i \\ \underline{b}_i \\ \underline{a}_{i+1} \\ \underline{b}_{i+1} \end{bmatrix} = \begin{bmatrix} \underline{v_1}^n(t_i) \\ \underline{v_1}^n(t_{i+1}) \end{bmatrix} ;\ i = 1, \dots, N-1 \tag{4.94}$$

- C^2 Kontinuitet: akselerasjon ved overgang fra et baneelement til neste skal være kontinuerlig dvs. $\underline{a^n}_{ai+1}(t_{i+1}) = \underline{a^n}_{ai+1}(t_{i+1})$. Dette gir 2(N-1) likninger.

$$\begin{bmatrix} A_{i+1} & \underline{0}^T & -A_{i+1} & \underline{0}^T \\ \underline{0}^T & A_{i+1} & \underline{0}^T & -A_{i+1} \end{bmatrix} \cdot \begin{bmatrix} \underline{a}_i \\ \underline{b}_i \\ \underline{a}_{i+1} \\ \underline{b}_{i+1} \end{bmatrix} = \underline{0} ;\ \ i = 1, \dots, N-1 \tag{4.95}$$

- Tilleggskrav: Det trenger to likninger for begge aksene til å definere akselerasjon i start og slutt punkter $\underline{a_1}^n(t_1)\ \ eller\ \ \underline{a_N}^n(t_{N+1})$

$$\begin{bmatrix} A_1 & \underline{0}^T \\ \underline{0}^T & A_1 \end{bmatrix} \cdot \begin{bmatrix} \underline{a}_1 \\ \underline{b}_1 \end{bmatrix} = \begin{bmatrix} \underline{v}^n(t_1) \\ \underline{a}^n(t_1) \end{bmatrix} = \begin{bmatrix} a_1{}^n(t_1) \\ a_2{}^n(t_1) \end{bmatrix} \tag{4.96}$$

Hvis det antas 3 baneelementer og 10 parametere så får vi 10x3 likninger.
Får å finne koeffisientene $\underline{\boldsymbol{a}}$ og $\underline{\boldsymbol{b}}$ settes posisjonsvektor, hastighetsvektor og akselerasjonsvektor sammen slik at vi får

$$\begin{bmatrix} A_1 & 0 & 0 \\ P_1 & 0 & 0 \\ P_2 & 0 & 0 \\ 0 & 0 & 0 \\ 0 & P_2 & 0 \\ 0 & P_3 & P_3 \\ 0 & 0 & P_4 \\ V_1 & 0 & 0 \\ V_2 & 0 & 0 \\ 0 & -V_2 & 0 \\ 0 & V_3 & 0 \\ 0 & 0 & V_3 \\ 0 & 0 & V_4 \\ A_2 & -A_2 & 0 \\ 0 & A_3 & -A_3 \end{bmatrix} \cdot \begin{bmatrix} \underline{a}_1 \\ \underline{a}_2 \\ \underline{a}_3 \end{bmatrix} = \begin{bmatrix} a_1{}^n(t_1) \\ p_1{}^n(t_1) \\ p_1{}^n(t_2) \\ p_1{}^n(t_2) \\ p_1{}^n(t_3) \\ p_1{}^n(t_3) \\ p_1{}^n(t_4) \\ v_1{}^n(t_1) \\ v_1{}^n(t_2) \\ v_1{}^n(t_2) \\ v_1{}^n(t_3) \\ v_1{}^n(t_3) \\ v_1{}^n(t_4) \\ 0 \\ 0 \end{bmatrix}, \begin{bmatrix} A_1 & 0 & 0 \\ P_1 & 0 & 0 \\ P_2 & 0 & 0 \\ 0 & 0 & 0 \\ 0 & P_2 & 0 \\ 0 & P_3 & P_3 \\ 0 & 0 & P_4 \\ V_1 & 0 & 0 \\ V_2 & 0 & 0 \\ 0 & -V_2 & 0 \\ 0 & V_3 & 0 \\ 0 & 0 & V_3 \\ 0 & 0 & V_4 \\ A_2 & -A_2 & 0 \\ 0 & A_3 & -A_3 \end{bmatrix} \cdot \begin{bmatrix} \underline{b}_1 \\ \underline{b}_2 \\ \underline{b}_3 \end{bmatrix} = \begin{bmatrix} a_1{}^n(t_1) \\ p_1{}^n(t_1) \\ p_1{}^n(t_2) \\ p_1{}^n(t_2) \\ p_1{}^n(t_3) \\ p_1{}^n(t_3) \\ p_1{}^n(t_4) \\ v_1{}^n(t_1) \\ v_1{}^n(t_2) \\ v_1{}^n(t_2) \\ v_1{}^n(t_3) \\ v_1{}^n(t_3) \\ v_1{}^n(t_4) \\ 0 \\ 0 \end{bmatrix} \tag{4.97}$$

Det kan skrives på følgende form

$$\underline{\boldsymbol{a}} = \boldsymbol{F^{-1}}\underline{\boldsymbol{h_1}}\ ,\ \underline{\boldsymbol{b}} = \boldsymbol{F^{-1}}\underline{\boldsymbol{h_2}} \tag{4.98}$$

- Naturlig 5. grads spline

I 5.grads naturlig spline kurve brukes det algoritmen som er beskrevet i avsnitt 4.2.1 polynombane. For *N* baneelementer trenger vi 6*N* likninger per akse. Dersom man spesifiserer posisjon, hastighet og akselerasjon i endepunkter for hver baneelement, kan det brukes polynom føringene for hver baneelement uavhengig av de andre baneelementer.

Da finner vi følgende parametere

$$\begin{bmatrix} P_i \\ P_{i+1} \\ V_i \\ V_{i+1} \\ A_i \\ A_{i+1} \end{bmatrix} \left[\underline{a_i}\right] = \begin{bmatrix} p_1^a(t_i) \\ p_1^a(t_{i+1}) \\ v_1^a(t_i) \\ v_1^a(t_{i+1}) \\ a_1^a(t_i) \\ a_1^a(t_{i+1}) \end{bmatrix}; \begin{bmatrix} P_i \\ P_{i+1} \\ V_i \\ V_{i+1} \\ A_i \\ A_{i+1} \end{bmatrix} \left[\underline{b_i}\right] = \begin{bmatrix} p_2^a(t_i) \\ p_2^a(t_{i+1}) \\ v_2^a(t_i) \\ v_2^a(t_{i+1}) \\ a_2^a(t_i) \\ a_2^a(t_{i+1}) \end{bmatrix} \quad for\ i = 1, \ldots . N \qquad (4.99)$$

4.3 Beskrivelse av MATLAB programmet

4.3.1 Sammensattbane

MATLAB programmet som er laget er bygget av en hoved program *'' Sammensatt bane''* som består av et antall funksjoner. Funksjonene består av bestemte algoritmer som er basert på matematisk utledninger omtalt i del 4.1 og 4.2 og er delt inn i ulike kategorier for å gi en oversikt over hvilke tilstander som må beregnes først. Hoved filen viser hvordan de ulike funksjonene henger sammen og hvilken sekvens som må startes først.

Brukeren av programmet kan selv velge ut hvilke start hastighet og akselerasjon som fartøyet skal kjøre med, om det er et fly eller undervannsfarkost. I tillegg kan man velge ut tiden det tar for hver baneelement. Dette gjøres manuelt i hoved programmet '' sammensatt bane'' ved å endre på inngangsverdier som er lagt inn i en matrise som kalles '' BD''.

Her er et eksempel på bane data '' BD'' matrise som inneholder 6 baneelementer hvor man starter og slutter med rettstrekning.

$$BD = \begin{bmatrix} 1 & T & V_0 & V \\ 1 & T & V & V \\ 2 & T & V & A \\ 3 & T & V & A \\ 4 & T & V & A \\ 1 & T & V & V \end{bmatrix}$$

Brukeren av programmet kan selv velge ut initialverdier. Første kolonne i BD matrise representerer banetype.

1- Rett strekning
2- Eulerspiral
3- Sirkel
4- Omvendt Eulerspiral

Hvor man har mulighet til å spesifisere hvilke bane vil kjøre ved å endre første kolonne. Andre kolonne foretar omløpstid '' *T* ''for hver baneelement. Tredje kolonne legges inn start hastighet '' *V*" og i fjerde kolonne ble det lagt inn start akselerasjon'' *A*''. Når det gjelder banetype 1 '' rett strekning'' ble det satt hastigheten i akselerasjonskolonne.

Viktige variabler som er interessante å få:

- Posisjon $\underline{p}^n$
- Hastighet $\underline{v}^n$
- Akselerasjon $\underline{a}^n$

Hvor *n* er navigasjonsramma f^n

Med initialverdier som er oppgitt i '' *BD* '' matrise, sendes disse til første funksjon ved et kall på funksjonen også videre i rekke følge.

Etter at alle gitte funksjonene er definert og plassert i riktig rekkefølge er banen klar til å generere. En for - løkke i sammensattbane utføres for at sekvensene i banen blir beregnet med riktig

matriseverdier og kaller på de ulike funksjonene som er definert. Funksjonene fullfører til sammen den komplette banen som er ønsket.

Bestemte algoritmer sørger for at fartøyet følger banen med kontinuerlig hastighet og akselerasjon. Sampeltid blir manuelt satt opp av brukeren. Underbaneregistrering lagres hvert posisjonssteg i en egen posisjonsmatrise og tiden lagres i en egen tidsmatrisen.

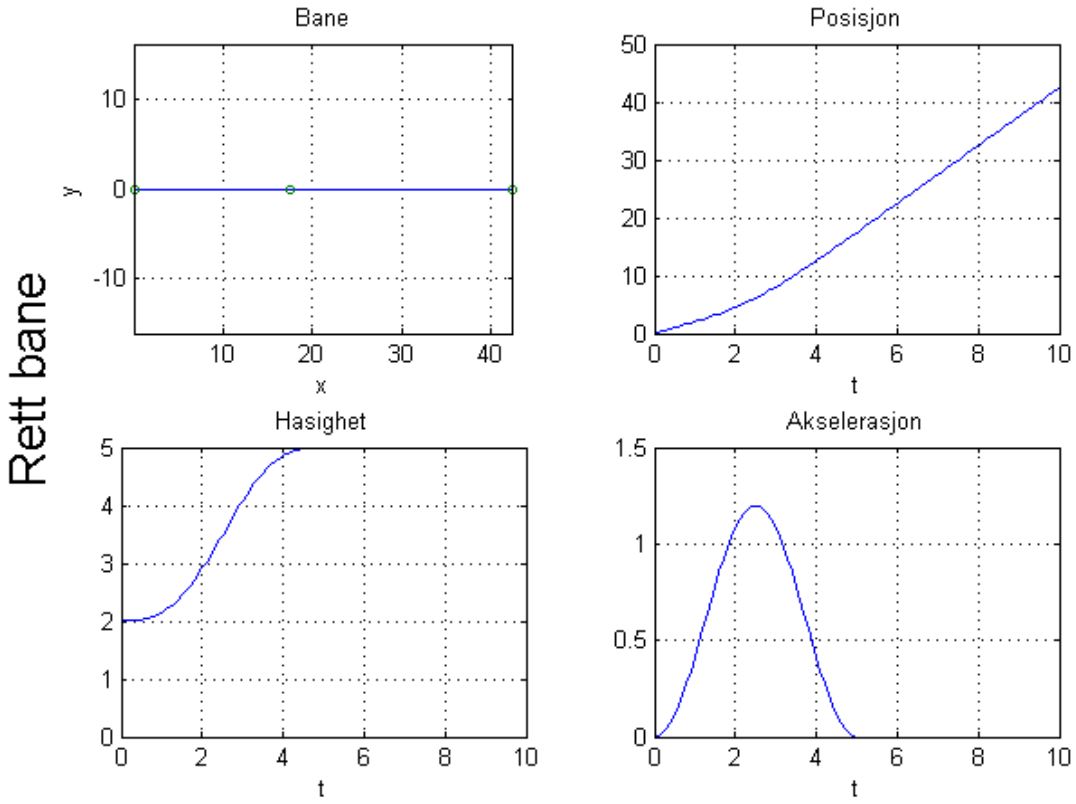
Med initialverdier som er oppgitt i matrise '' BD'' får vi bestemteverdier av skjøtepunkter mellom baneelementer.

Disse skjøtepunkter kalles wp som er endepunkter i $\underline{P^n}$ matrise. De vil brukes videre som initialverdier for andre bane algoritmer som er beskrevet av Spline metode. Denne bane kan brukes som en referanse bane og vil sammenlignes med andre metoder i kap.5.

For å få posisjonen, akselerasjon og hastighet i det globale ramma for banen f^n brukes rotasjonsmatrise R_3. Det ble lagt en egen funksjon som foretar beregning av rotasjonsmatrise som kalles R_3.m. Ved innkalling av denne funksjonen får vi rotere posisjon, hastighet og akselerasjon fra lokal ramme f^a til global ramme f^n.

Nå som alle funksjonene er definert, vil programmet ha en oversikt over alle variablene som er beregnet. Ved å endre på verdiene av BD matrise, vil hele programmet foreta nye beregninger som er annerledes enn det som tidligere ble estimert. Banen vil konstrueres på en annen måte og svingene vil bli endret. Når en del kjøres, vil det utføres beregninger og kommer med nye oppdaterte verdier.

4.3.2 *RBA*.m.funksjon



Figur (4-3) RBA.m funksjon

Figur(4-3) viser et eksempel på bruk av RBA.m funksjon som foretar beregning av rettstrekning. Hvor det antas to baneelementer. I det første baneelementet antas det at fartøyet starter bevegelse med å øke hastigheten fra 2 til 5 m/s, og fortsetter konstant i det andre baneelementet.

Pseudokode for *RBA*.m funksjon

$$function\left[\underline{p}^a, \underline{v}^a, \underline{a}^a\right] = RBA(t, T, V_0, V_1)$$

$$\omega = \frac{2\pi}{T}\ ; A = \frac{V_1 - V_0}{T};$$

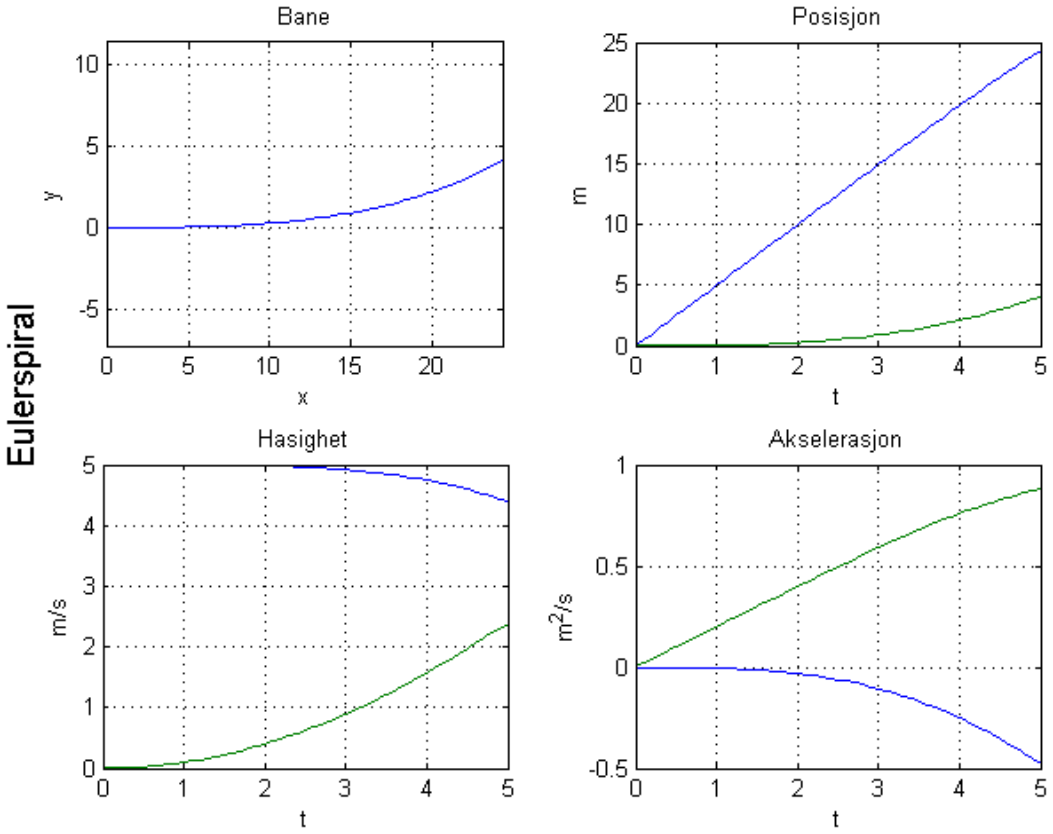
$$\underline{p}^a(t) = \begin{bmatrix} V_0\, t + \frac{1}{2}At^2 - A/\omega^2(1 - cos\omega t) \\ 0 \end{bmatrix}$$

$$\underline{v}^a(t) = \begin{bmatrix} V_0 + A(t - \frac{1}{\omega} sin\omega t \\ 0 \end{bmatrix}$$

$$\underline{a}^a(t) = \begin{bmatrix} A(t - cos\omega t \\ 0 \end{bmatrix}$$

4.3.3 ES.m funksjon

ES.m funksjonen tar i utgangspunkt beregning av Eulerspiral som representerer baneelementet i området mellom rettlinje og sirkelbue.



Figur (4-4) ES.m funksjon

Figur (4-4) viser et eksempel på ES.m funksjon, hvor første plott viser banen. Andre plott viser posisjon i forhold til tiden. Den tredje viser hastighet hvor retning x mister pådraget sitt etter ca. 2 sekunder og styres av y- retning. Vi ser også at akselerasjon avtar gradvis og styres av y-aksen, hvor blå linje representere x-retning og grønnlinje representerer y-aksen.

Pseudokode for ES.m funksjon

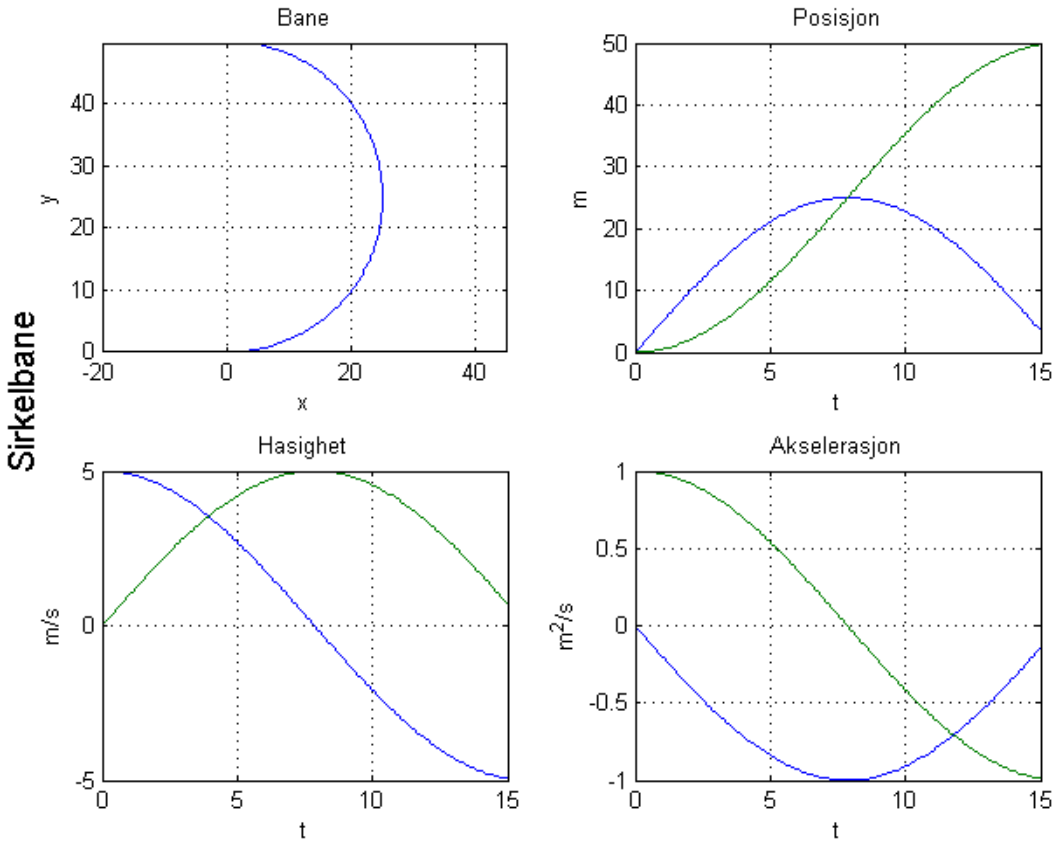
$$function\left[\underline{p}^a, \underline{v}^a, \underline{a}^a\right] = RBA(t, T, V, A)$$

$$\theta(t) = \frac{At^2}{2VT};\ \theta(T) = \frac{AT}{2V};\ \Omega = \sqrt{\frac{A}{\pi VT}}$$

$$\underline{p}^a(t) = \frac{V}{\Omega}\begin{bmatrix} C(\Omega t) \\ S(\Omega t) \end{bmatrix};\ \underline{v}^a(t) = V\begin{bmatrix} cos\theta(t) \\ sin\theta(t) \end{bmatrix};\ \underline{a}^a(t) = At/T\begin{bmatrix} -sin\theta(t) \\ cos\theta(t) \end{bmatrix}$$

4.3.4 SB.m funksjon

SB.m representerer banen i sirkelbevegelse som er tegnet i det første plottet. For å teste disse funksjonene settes start punkt lik null mens i hoved programmet kjøres disse funksjonene i sekvens slik som vist i '' BD'' matrise.



Figur (4-5) SB.m funksjon

Pseudokode for SB.m funksjon

$$function\left[\underline{p}^a, \underline{v}^a, \underline{a}^a\right] = SB(t, T, V, A)$$

$$\omega = \frac{A}{V};\ \theta(t) = \omega t;$$

$$\underline{p}^a(t) = R\begin{bmatrix} sin\theta \\ 1 - cos\theta \end{bmatrix};$$

$$\underline{v}^a(t) = V\begin{bmatrix} cos\theta \\ sin\theta \end{bmatrix};\ \underline{a}^a(t) = A\begin{bmatrix} -sin\theta \\ cos\theta \end{bmatrix};$$

4.3.5 OES.m

Baneelementet '' omvendt Eulerspiral '' ble definert ved funksjonen *OES.m*. hvor fartøyet kjører i motsatt retning fra sirkelbevegelse til rettstrekning. Her brukes det samme algoritmen for Eulerspiral men i dette tilfelle settes tiden $t = (T - t)$

Pseudokode for OES.m funksjon

$$function\left[\underline{p}^a, \underline{v}^a, \underline{a}^a\right] = OES(t, T, V, A)$$

$$\theta(t) = \frac{A}{V\left(t - \frac{t^2}{2T}\right)};\ \theta(T) = \frac{AT}{2V};\ \Omega = \sqrt{\frac{A}{\pi VT}};$$

$$\underline{p}^a(t) = \frac{V}{\Omega}\begin{bmatrix} -cos\theta T & -sin\theta T \\ -sin\theta T & cos\theta T \end{bmatrix}\begin{bmatrix} C(\Omega(T-t) - C(\Omega T) \\ S(\Omega(T-t)) - S(\Omega T) \end{bmatrix};$$

$$\underline{v}^a(t) = V\begin{bmatrix} cos\theta(t) \\ sin\theta(t) \end{bmatrix};\ \underline{a}^a(t) = \frac{A(T-t)}{T}\begin{bmatrix} -sin\theta(t) \\ cos\theta(t) \end{bmatrix};$$

4.3.6 PB.m funksjon

PB.m funksjonen foretar beregning av polynomligning. Det antas at man kjenner start og slutt verdi av tid, hastighet og akselerasjon. Beregningene er beskrevet i form av matematisk vektorer og likninger i avsnitt (4.2.1).

Det vil bruke anonyme funksjoner ved hjelp av operatoren @ til å definere parametrisk ligninger for posisjon, hastighet og akselerasjon.

Pseudokode for PB.m funksjon er skrevet nedenfor

$function\left[\underline{p}^{a}, \underline{v}^{a}, \underline{a}^{a}\right] = PB(t, t_1, t_2, \underline{P_1}^{n}, \underline{P_2}^{n}, \underline{v_1}^{n}, \underline{v_2}^{n}, \underline{a_1}^{n}, \underline{a_2}^{n})$

$P(t) = [\ t^5\ \ t^4\ t^3\ \ t^2\ \ t\ \ 1\ \ ]\ ;\ P_1 = P\ (t_1); P_2 = P\ (t_2);$

$V(t) = [\ \ 5t^4 4t^3\ \ 3t^2\ \ 2\ t\ \ 1\ \ 0\ \ ]\ ;\ V_1 = V\ (t_1);\ V_2 = V\ (t_2);$

$A(t) = [20t^3\ \ 12t^2\ \ 6t\ \ 2\ \ 0\ \ 0]\ ;\ \ A_1 = A\ (t_1); A_2 = A\ (t_2);$

Løser vi ligning (4.71)

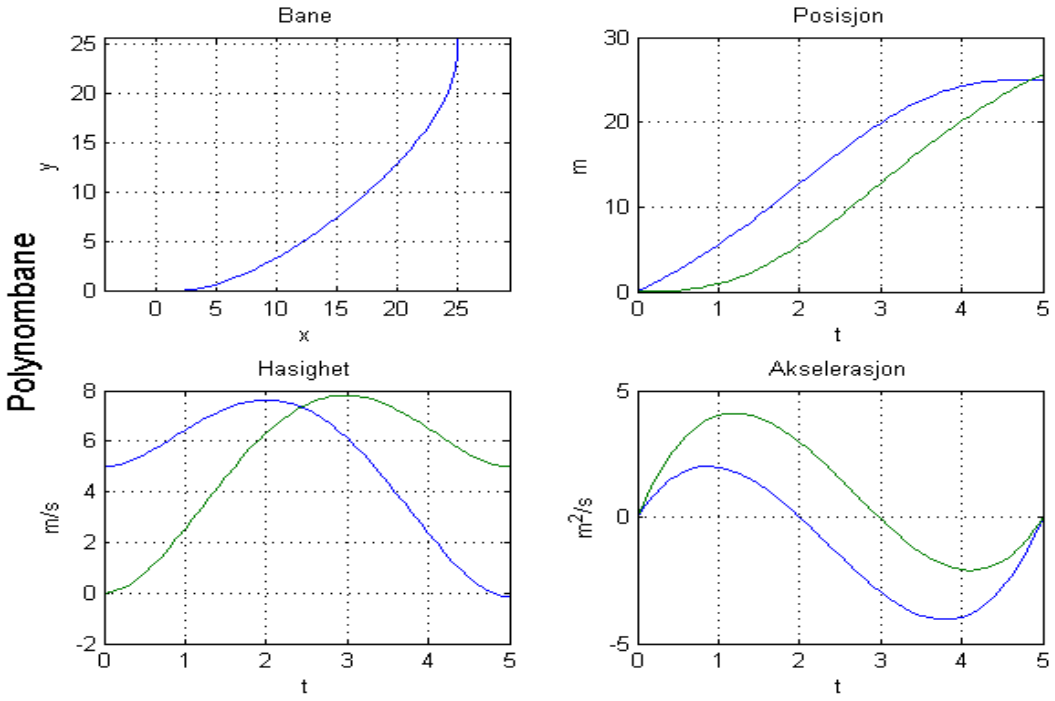
$\underline{p}^{a}(t) = \left[P(t)\underline{a};\ P(t)\underline{b}\right];$

$\underline{v}^{a}(t) = \left[V(t)\underline{a};\ V(t)\underline{b}\right];$

$\underline{a}^{a}(t) = \left[A(t)\underline{a};\ A(t)\underline{b}\right];$

For å teste om funksjonen fungerer slik som vist i figuren (4-6), antas det følgende inngangsverdier

$$\begin{bmatrix} P_1 \\ P_2 \\ V_1 \\ V_2 \\ A_1 \\ A_2 \end{bmatrix} = \begin{bmatrix} 0 & 0 \\ 46.126 & 49.443 \\ 5 & 0 \\ 0 & 5 \\ 0 & 0 \\ 0 & 0 \end{bmatrix}$$



Figur (4-6) PB.m funksjon

Med de valgte inngangsverdier ser vi at polynom funksjon kan brukes til å representer banen siden kravet for kontinuitet i posisjon, hastighet og akselerasjon er oppfylt.

4.3.7 SP3B.m funksjon

Her lages et program som representerer banen ved å bruke '' Naturlig Spline metoden'', hvor det antas tredjegrads naturlig spline kurve, som det ble beskrevet i avsnitt 4.2.2. Det antas at vi kjenner følgende verdier

- Start hastighet og akselerasjon
- Endepunkter for hver baneelement (veipunkter)
- Tiden i endepunkter for hver baneelement

Funksjonen tar i utgangspunkt initialverdier som er brukt i sammensattbane.m programmet. Deretter lagrer parameter likningene ved å lage anonyme funksjoner ved hjelp av operatoren @.
Koeffisientene beregnes slik som tidligere har nevnt i avsnitt 4.2.2.

Pseudokode for SP3B.m funksjon er skrevet nedenfor

$$function\left[\underline{p}^n(t), \underline{v}^n(t), \underline{a}^n(t)\right] = SP3B(t, \underline{t}^{wp}, P^{wp}, \underline{v_1}^n, \underline{a_1}^n)$$

$$P(\tau) = [\,\tau^3 \quad \tau^2 \quad \tau \quad 1\,];\ P_i = P\,(t_i);$$

$$V(\tau) = [\,3\tau^2 \quad 2\tau \quad 1 \quad 0\,];\ V_i = V\,(t_i);$$

$$A(t) = [\,6\tau \quad 1 \quad 0 \quad 0];\ A_i = A\,(t_i);$$

Løser vi ligning (4.84) og (4.86)

$$\underline{p}^n(t) = \left[P(t)\underline{a};\ P(t)\underline{b}\right]; for\ t\epsilon\left[t_i^{wp}, t_{i+1}^{wp}\right], i = 1{:}\,N$$

$$\underline{v}^n(t) = \left[V(t)\underline{a};\ V(t)\underline{b}\right];\ for\ t\epsilon\left[t_i^{wp}, t_{i+1}^{wp}\right], i = 1{:}\,N$$

$$\underline{a}^n(t) = \left[A(t)\underline{a};\ A(t)\underline{b}\right]; for\ t\epsilon\left[t_i^{wp}, t_{i+1}^{wp}\right], i = 1{:}\,N$$

4.3.8 SP4B.m funksjon

Det ble laget en egen funksjons fil som utfører beregning av Naturlig 4.grads algoritmen. Funksjonen tar i utgangspunkt SP3B.m funksjonen.

Pseudokode for SP4B.m funksjon er skrevet nedenfor

$function\left[\underline{p}^n(t), \underline{v}^n(t), \underline{a}^n(t)\right] = SP4B(t, \underline{t}^{wp}, P^{wp}, V^{wp}, \underline{a_1}^n)$

$P(\tau) = [\ \tau^4 \quad \tau^3 \quad \tau^2 \quad \tau \quad 1\ ];\ P_i = P\ (t_i);$

$V(\tau) = [\ 4\tau^3 \quad 3\tau^2 \quad 2\tau \quad 1 \quad 0\ ]\ ;\ V_i = V\ (t_i);$

$A(t) = [12\tau^2 \quad 6\tau \quad 1 \quad 0 \quad 0]\ ;\ A_i = A\ (t_i);$

Løser vi ligning (4.98)

$\underline{p}^n(t) = \left[P(t)\underline{a};\ P(t)\underline{b}\right]; for\ t\epsilon\left[t_i^{wp}, t_{i+1}^{wp}\right], i = 1{:}\ N$

$\underline{v}^n(t) = \left[V(t)\underline{a};\ V(t)\underline{b}\right];\ for\ t\epsilon\left[t_i^{wp}, t_{i+1}^{wp}\right], i = 1{:}\ N$

$\underline{a}^n(t) = \left[A(t)\underline{a};\ A(t)\underline{b}\right]; for\ t\epsilon\left[t_i^{wp}, t_{i+1}^{wp}\right], i = 1{:}\ N$

4.3.9 SP5B.m funksjon

Pseudokode for SP5B.m funksjon er skrevet nedenfor

$$function\left[\underline{p}^n(t), \underline{v}^n(t), \underline{a}^n(t)\right] = SP5B(t, \underline{t}^{wp}, P^{wp}, V^{wp}, A^{wp})$$

$$P(\tau) = [\ \tau^5 \ \ \tau^4 \ \ \tau^3 \ \ \tau^2 \ \ \tau \ \ 1\ ];\ P_i = P\ (t_i);$$

$$V(\tau) = [\ \ 5\tau^4 \ 4\tau^3 \ \ 3\tau^2 \ 2\tau \ \ 1 \ \ 0\ ]\ ;\ V_i = V\ (t_i);$$

$$A(t) = [\ 20\,\tau^3 \ \ 12\tau^2 \ \ 6\tau \ \ 1 \ \ 0 \ \ 0]\ ;\ \ A_i = A\ (t_i);$$

Løser vi ligning (4.99)

$$\underline{p}^n(t) = \left[P(t)\underline{a};\ P(t)\underline{b}\right]; for\ t\epsilon\left[t_i^{wp}\ , t_{i+1}^{wp}\ \right], i = 1{:}\ N$$

$$\underline{v}^n(t) = \left[V(t)\underline{a};\ V(t)\underline{b}\right];\ for\ t\epsilon\left[t_i^{wp}\ , t_{i+1}^{wp}\ \right], i = 1{:}\ N$$

$$\underline{a}^n(t) = \left[A(t)\underline{a};\ A(t)\underline{b}\right]; for\ t\epsilon\left[t_i^{wp}\ , t_{i+1}^{wp}\ \right], i = 1{:}\ N$$

5. Simuleringsresultater

Fremgangsmåten og beregninger som er gjort rede for, er testet og simulert i henhold til begrunnet teoretisk bakgrunn og viser resultater for hele systemet som er blitt implementert med bestemte algoritmer i MATLAB. De ulike delene som fremstiller resultatene, viser grafiske figurer som beskriver tilstandene for hver bane element.

Simuleringsresultatene for baneelementer ønskes å være brukbare slik at de kan konkluderes på en logisk måte. Sammensatt baneelementer som er beskrevet i form av rettstrekning, Eulerspiral og sirkelbevegelse brukes som referanse bane, dermed sammenliknes tilstandsverdier fra de ulike algoritmer som er testet med referanse bane.

Figurene er fremstilt med x- og y- akser, hvor hver akse har sin egen definert beskrivelse. Akse y gjengir fartøyet tilstand i $m, m/s, m/s^2$. Akse x beskriver fartøyet med hensyn til tiden i sekunder.

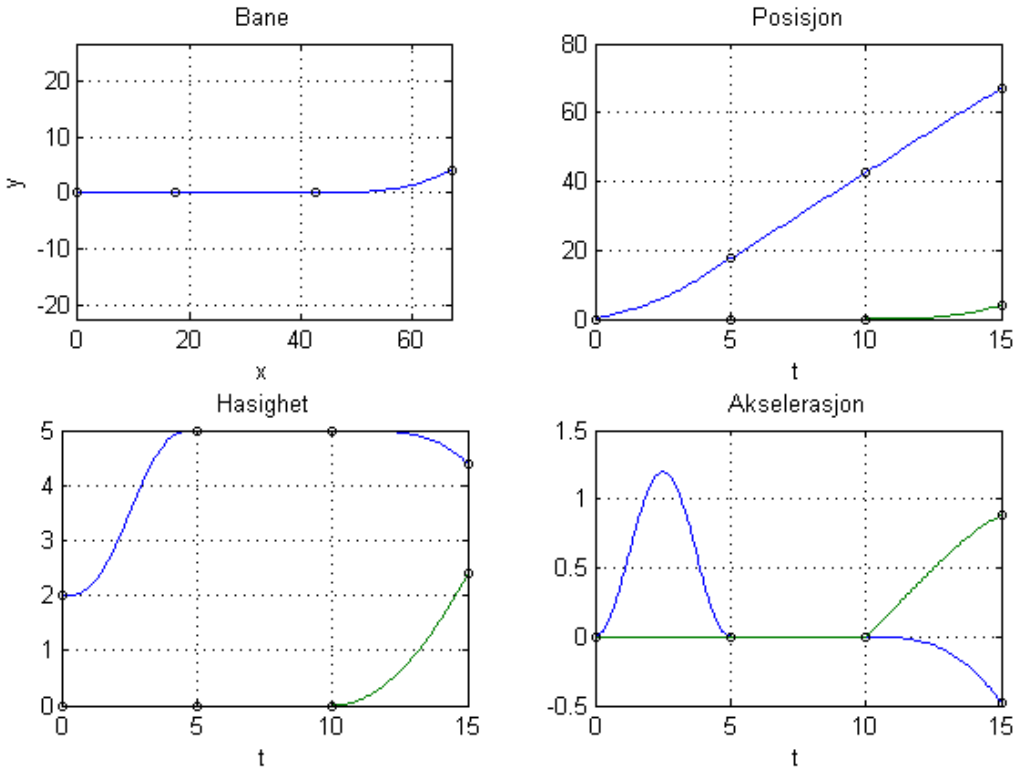
Hver figur vil bli beskrevet og diskutert i henhold til hva som er oppnådd, og enkelte vil bli begrunnet dersom eventuelle resultater ikke ble som ønsket.

5.1 Eulerspiral

- Referansebane

Realisering av banen er fremstilles i figur (5.1). Følgelig lages en 2 dimensjonal koordinatsystem som illustrerer fartøys gjennomkjøring i banen sett fra {n} ramma. Det vil starte med å presentere de tre første baneelementer i referansebane.

1. Plott 1 viser banen med 3 baneelementer rettstrekning med akselerasjon, rettstrekning uten akselerasjon og Eulerspiral. Baneelementer er koblet sammen med skjøtepunkter '' o''.
2. Andre plott viser posisjon som starter fra null og øker lineart i x-retning som det skal.
3. Tredje plott viser hastighet som øker fra 2 til 5 m/s i det første baneelement, deretter blir konstant når akselerasjonen settes lik null. Retning x mister pådraget sitt gradvis når en sving inntreffer og styres i y-retning.
4. Det fjerde plottet viser akselerasjon som øker litt over 1 m^2/s i det første baneelement så kommer tilbake til null. I svingen får x en gradvis negativ påvirkning, mens y får lineart positiv påvirkning.



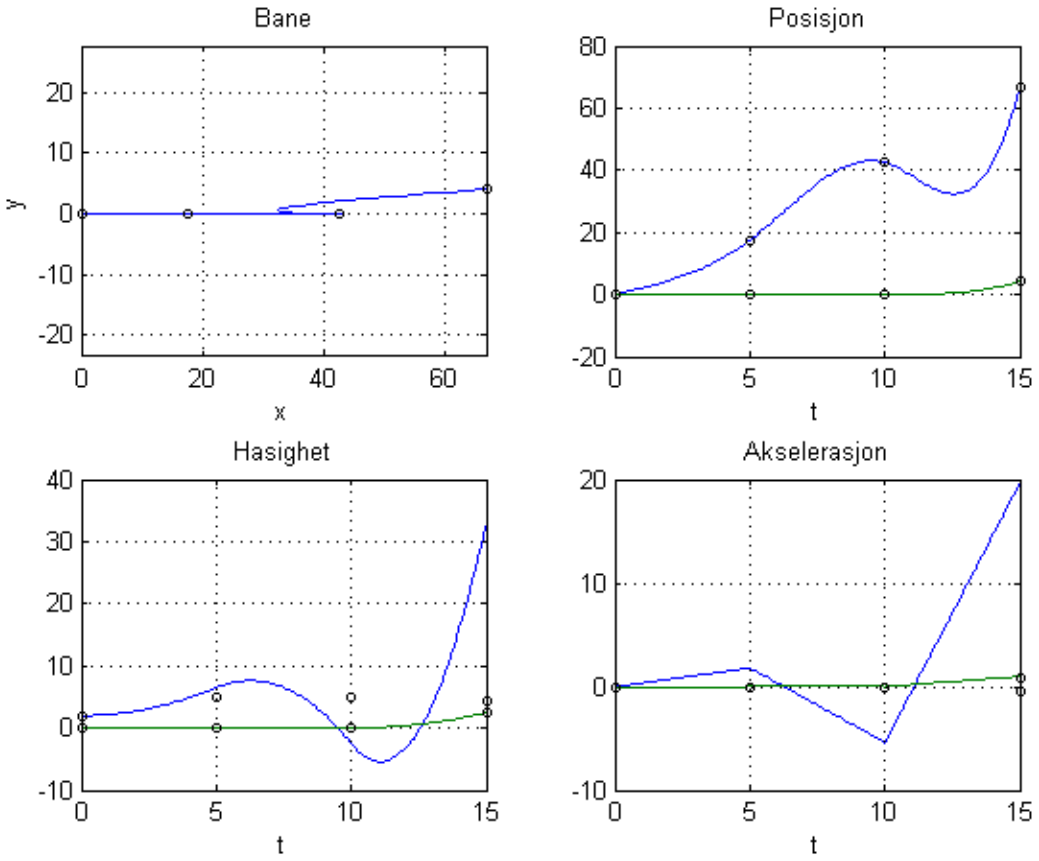
Figur (5-1) sammensattbane (Eulerspiral)

Fra figuren (5-1) ser vi at denne bane gir oss et tilfredsstillende resultat dermed kan den brukes som referansebane og vil sammenliknes med spline metoden.

- 3.grads naturlig spline bane

Simuleringsresultatet for 3.grads spline bane samt posisjon, hastighet og akselerasjon er vist i figur (5-2) nedenfor.

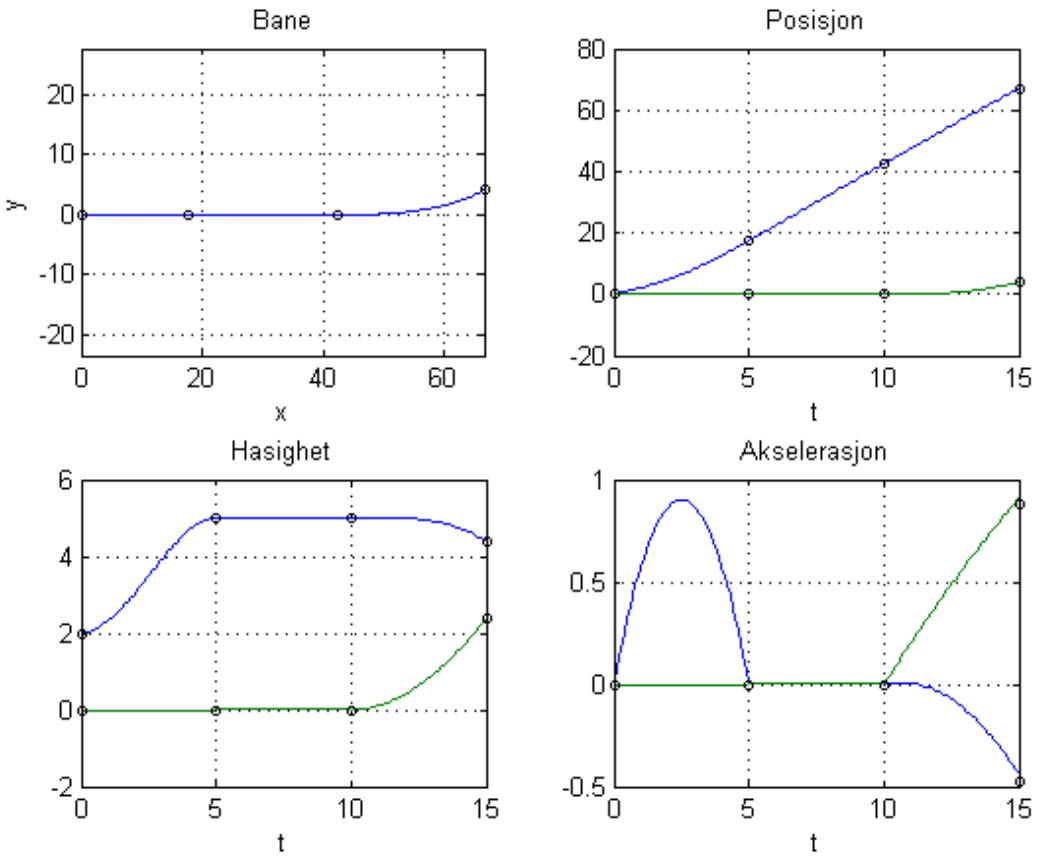
Vi ser at banen i det første plottet plasseres ut av kurset i forhold til referanse bane når den inntreffer en sving. Dette kan sees betydelig fra posisjon plott hvor x-retning mister pådraget sitt i det tredje baneelementet. Hastighet og akselerasjon plottene er ikke særlig realistisk, og dermed vil det konkludere at 3.grads naturlig spline bane er ikke en brukbar bane.



Figur (5-2) 3.gradsnaturlig splinebane (sammenliknet med Eulerspiral)

- 4.grads naturlig spline bane

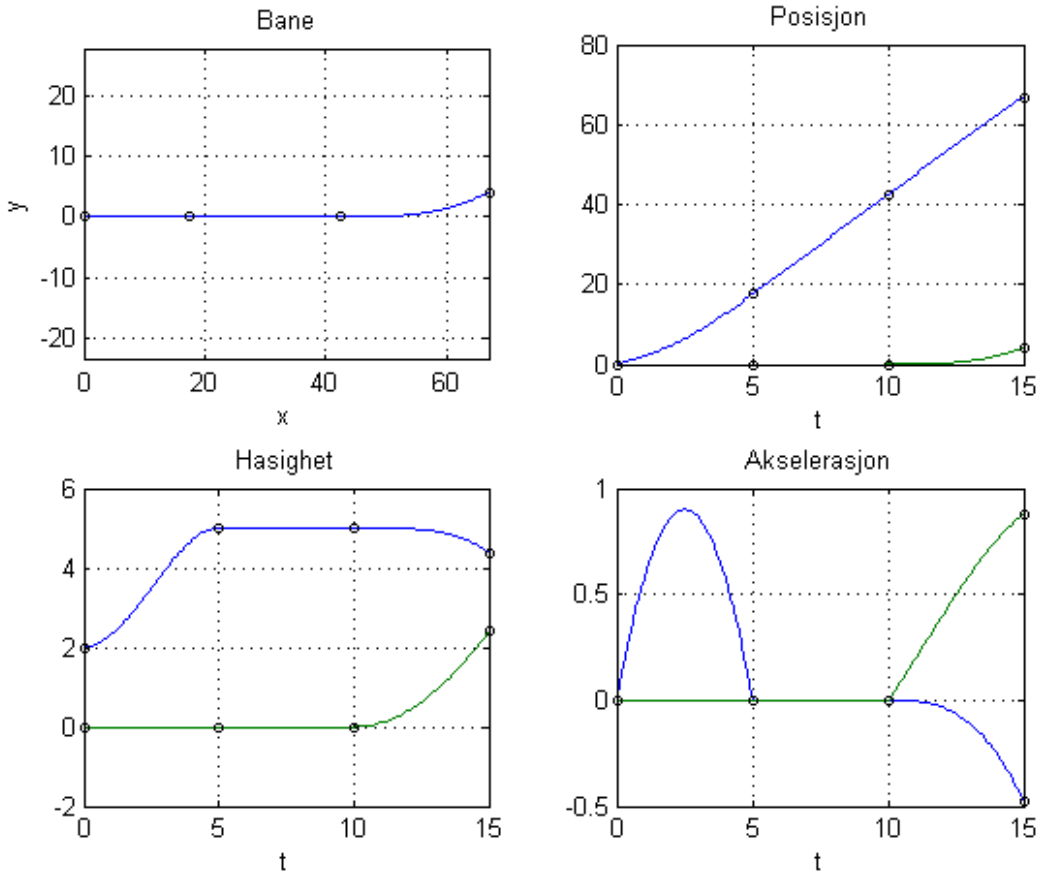
Siden 3.grads naturlig spline kurven ikke gir oss et tilfredsstillende resultat, økes graden til 4.ordens spline. Det første plottet i figuren (5-3) viser simuleringsresultat for banen. Banen plasseres riktig i forhold til referanse bane. Andre plott viser posisjon hvor x-retning øker lineart som det skal. Hastighet og akselerasjon stemmer overens med resultatet trukket ut fra referanse bane.



Figur (5-3) 4.gradsnaturlig splinebane (sammenliknet med Eulerspiral)

- 5.grads naturlig spline bane

Figur (5.4) viser simuleringsresultat for 5.grads spline bane som tar utgangspunkt i beregningen som er utført i 5.grads polynom algoritmen.



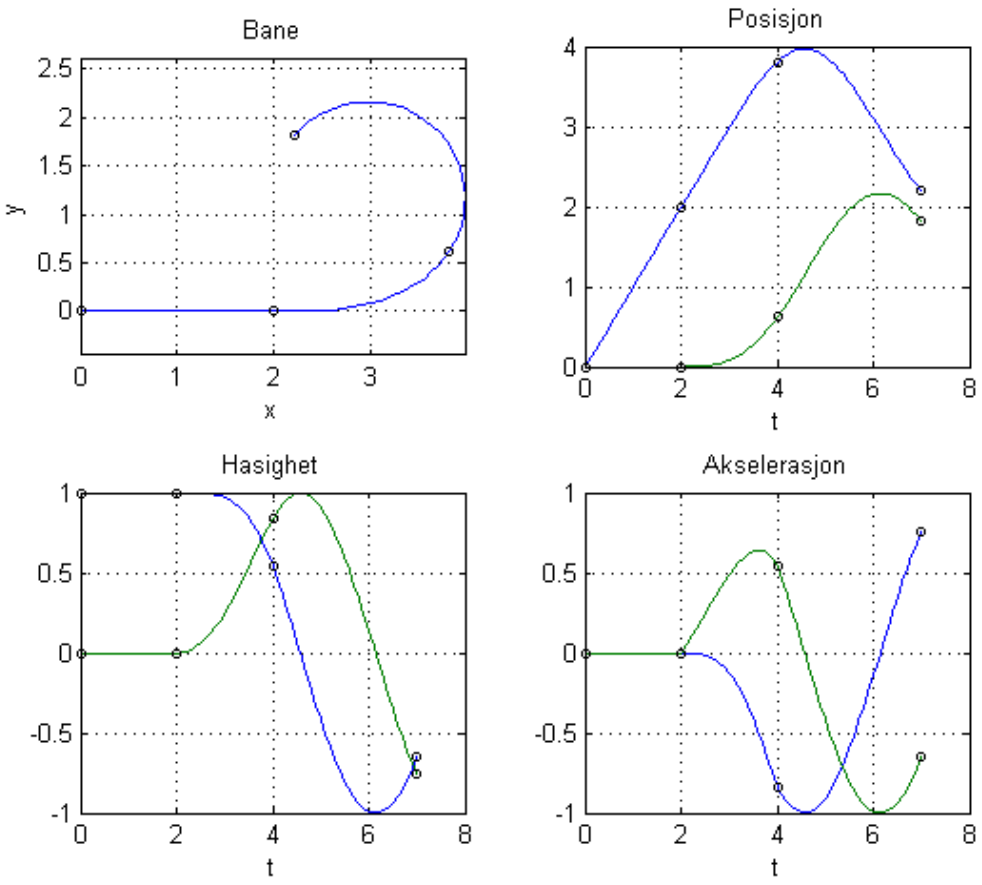
Figur (5-4) 5.gradsnaturlig splinebane (sammenliknet med Eulerspiral)

Resultatene har tilnærmet like verdier som referanse bane. Ved å sammenlikne figur(5.4) og (5.3) ser man dette stemmer, og verdiene for bane, posisjon, hastighet og akselerasjon er som ønsket. Det kan konkluderes med at Eulerspiral kan erstattes med 4 og 5.grads naturlig spline algoritmer.

5.2 Bane i svingebevegelse

- Referansebane

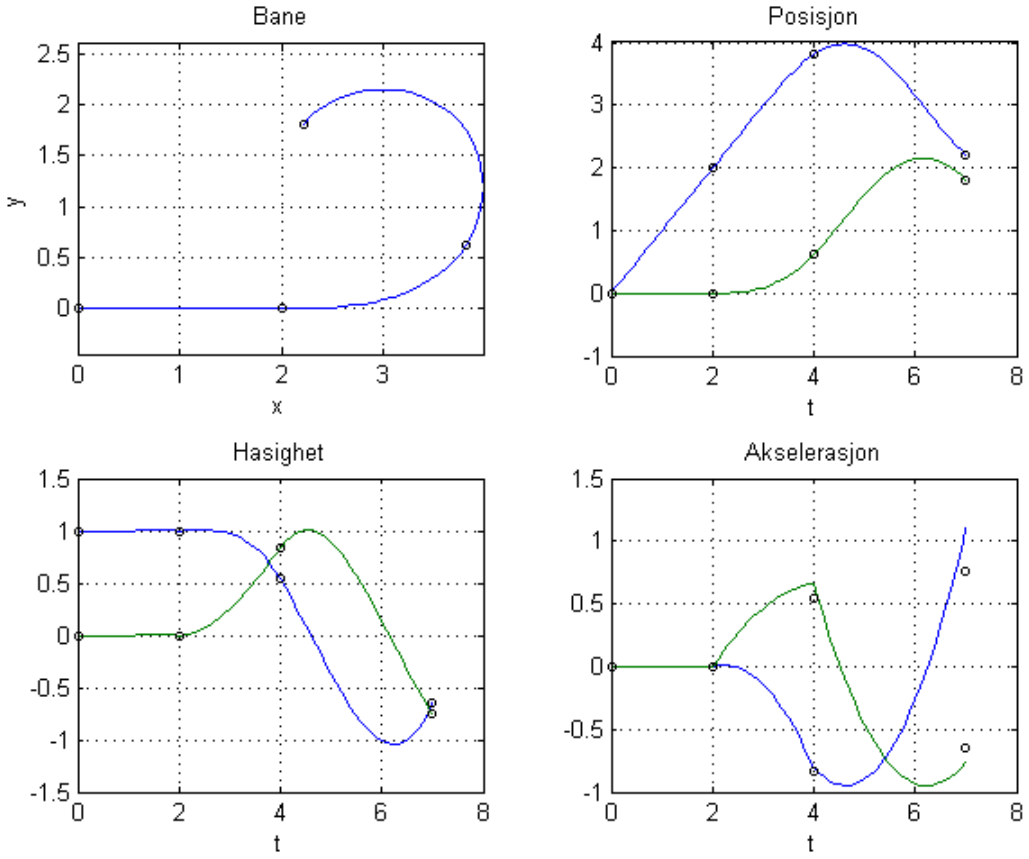
Her testet vi tre baneelementer rettstrekning, Eulerspiral og sirkelbevegelse i rekkefølge. Dette er vist i plott 1 av figur(5.5). Plott 2 viser posisjonen som øker gradvis i x-retning i de to første baneelementer, så avtar det når fartøyet snur i motsatt retning som det skal. Hastigheten i plott 3 er konstant på 1 m/s i det første baneelementet og styres av x-retning, så avtar til 0.5 i overgang mellom rettlinje og sirkel deretter fortsetter nedover til å utforme en halv sirkel i det siste baneelementet. Plott 4 tar utgangspunkt akselerasjon som avtar gradvis i Eulerspiral og styres av y-retning til at fartøyet kommer til en sirkel bevegelse så begynner x-retning å vise positiv påvirkning. Dette viser en optimal løsning siden banen gir kontinuitet i hastighet og akselerasjon.



Figur (5-5) Sammensattbane i svingebevegelse

- 4.grads naturlig spline bane

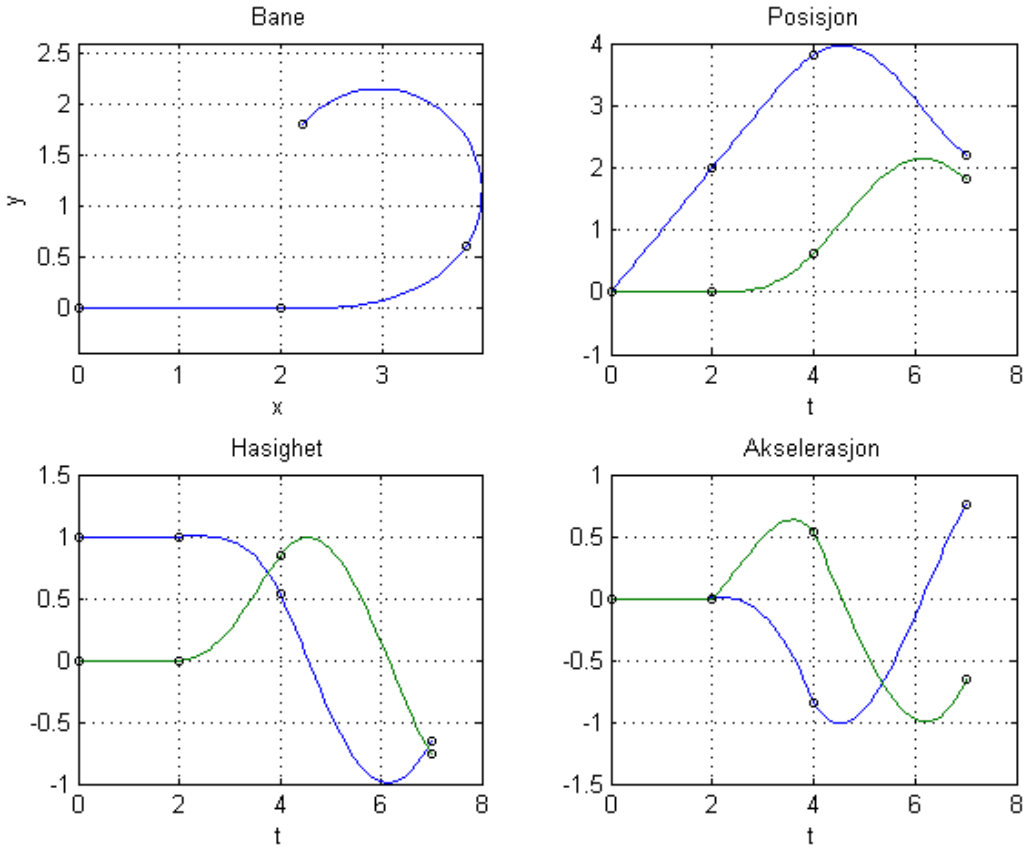
Simuleringsresultat for 4.ordens spline bane er vist i figur (5.6). Resultatet er tilnærmet lik det som er skissert i referanse bane, med unntak fra akselerasjonsplott hvor det er en sprang i y-retning ved overgang fra andre til tredje baneelement. Ellers ser tilstandene for posisjon, hastighet og akselerasjon kontinuerlige som ønsket.



Figur (5-6) 4.gradsnaturlig splinebane i svingebevegelse

- 5.grads naturlig spline bane

5.grads spline bane, viser til simuleringsresultatet under.



Figur (5-7) 5.gradsnaturlig splinebane i svingebevegelse

Man ser tydelig at simuleringsresultatet er som forventet. Kravet for kontinuitet i posisjon, hastighet og akselerasjon er oppfylt. Dermed vil banen med 5.grads naturlig spline algoritmen betraktes som en tilfredsstillende bane.

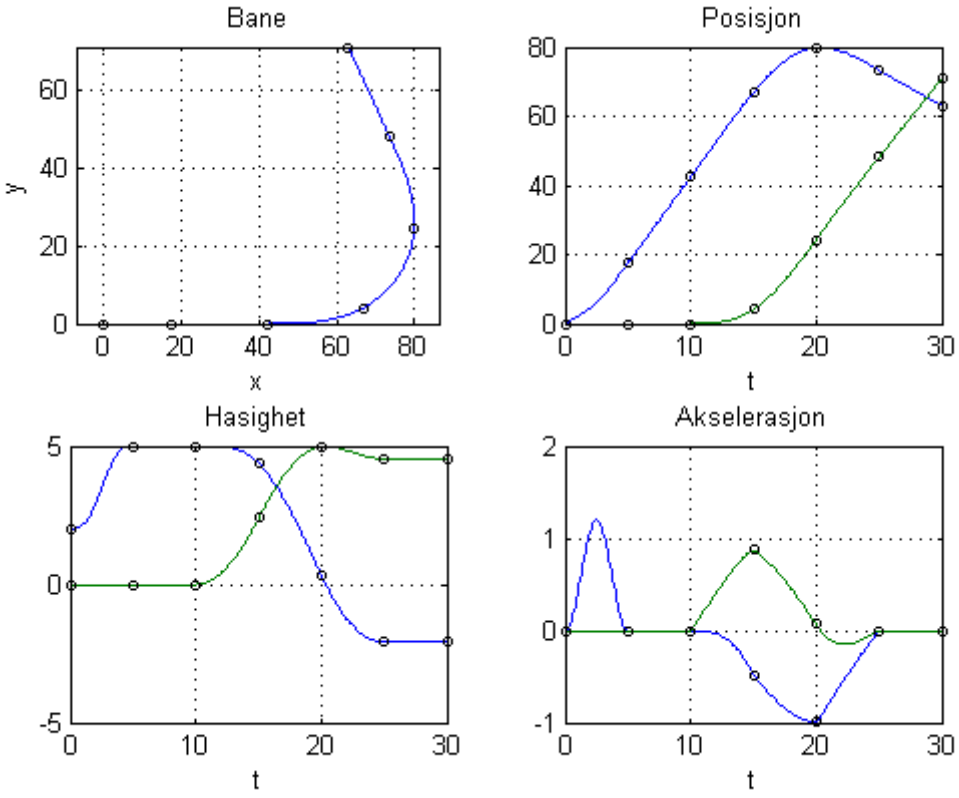
5.3 Sammensatt av baneelementer

Banen som er fremstilt i det første plott av figur (5.8) viser 6 baneelementer. Her presenteres de matematiske beregning som er utført i form av forskjellige type baneelementer. Andre plott vil vise posisjonen som starter fra null og øker lineart så avtar i x-retning og blir styrt av y–retning ved overgang fra sirkel til omvendt Eulerspiral.

Tredje plott viser hastighet som øker fra 2 til 5 m/s i det første baneelement, deretter blir konstant når akselerasjonen settes lik null. Retning x mister pådraget sitt når en sving inntreffer og styres i y-retning helt til siste baneelement.

Det fjerde plottet viser akselerasjon som utformer en cosinus funksjon i det første baneelement, så settes lik null i andre baneelementet. I svingen får x en gradvis negativ påvirkning, mens y får positiv påvirkning og reduseres gradvis ned så settes det lik null i rett bevegelse.

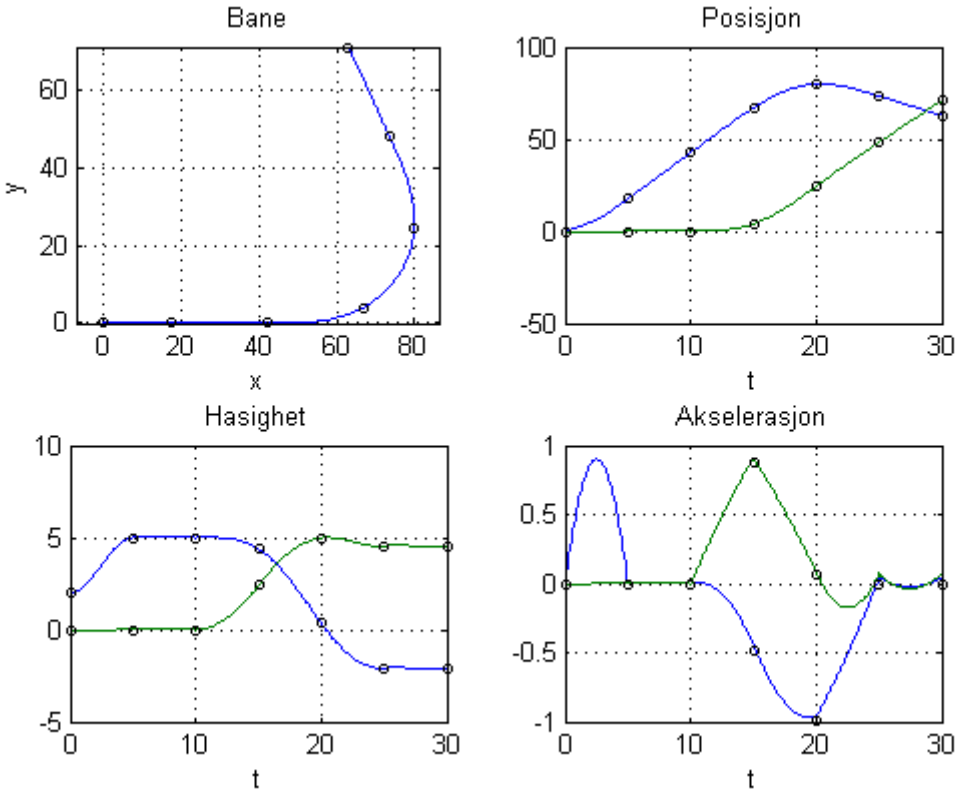
- Referansebane



Figur (5-8) Sammensattbane

Det skal fortsette å sammenlikne referanse bane som er vist i figuren (5-8) med bane representert av spline metode.

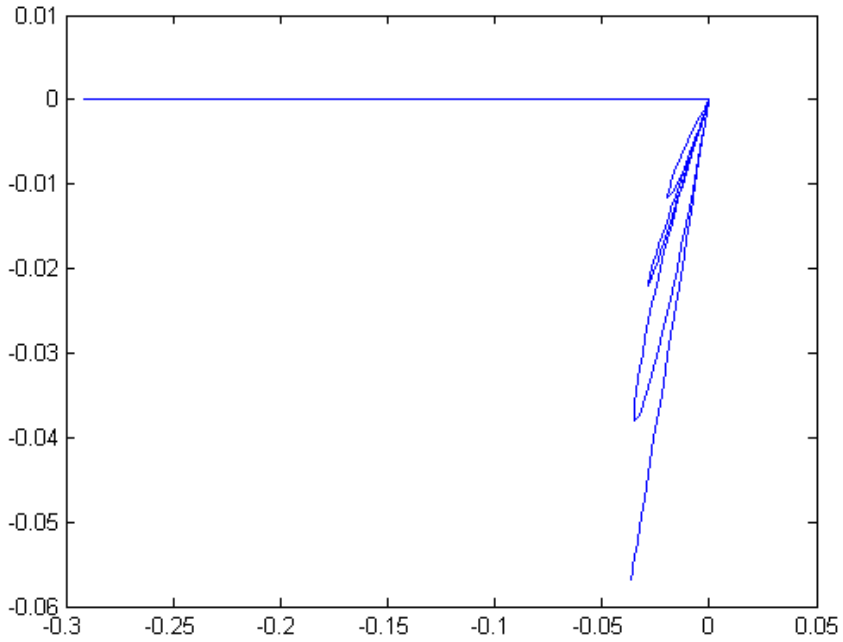
- 4.grads naturlig spline bane



Figur (5-9) 4.gradsnaturlig splinebane

Det første plottet i figuren (5-9) viser simuleringsresultat for banen. Banen plasseres ganske riktig i forhold til referanse bane. Andre plott viser posisjon hvor x-retning øker lineart som det skal. Når det gjelder hastighet så vil den stemme i forhold til referanse bane. Akselerasjon i det første baneelementet ser ut som en parabel mens i referanse bane er den utformet som en cosinus funksjon. I den siste delen ligger akselerasjon ut av kurset.

Det er tatt differanse mellom referanse bane og 4.grads spline som er skissert i figuren (5-10).

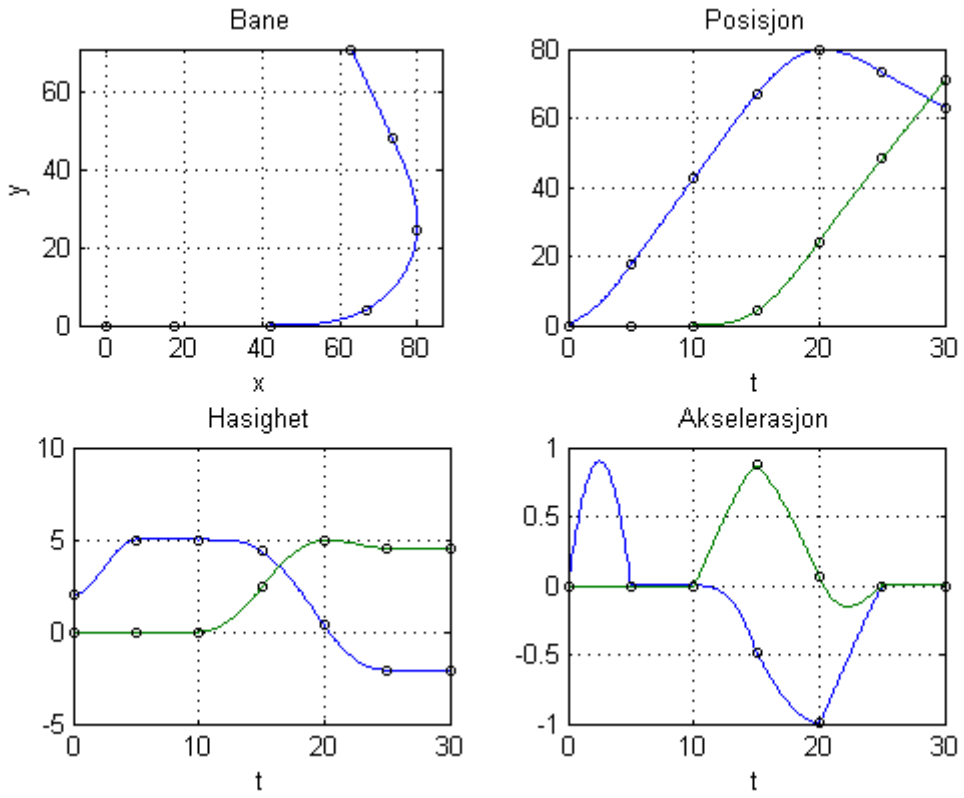


Figur (5.10) differanse mellom referanse bane og 4.grads naturlig spline bane

Støyen som forplanter seg i verdiene er vist i figuren (5.10). Det kan være to mulige grunner med at spline metode er bygd av polynom likning, som det er vist i akselerasjonsplottet hvor første baneelementet utformer en parabel, mens i referanse bane utformer en cosinus funksjon. I tillegg til numeriske feil siden spline er en numerisk metode. Dette vil lede til å teste 5.grads spline bane.

- 5.grads naturlig spline bane

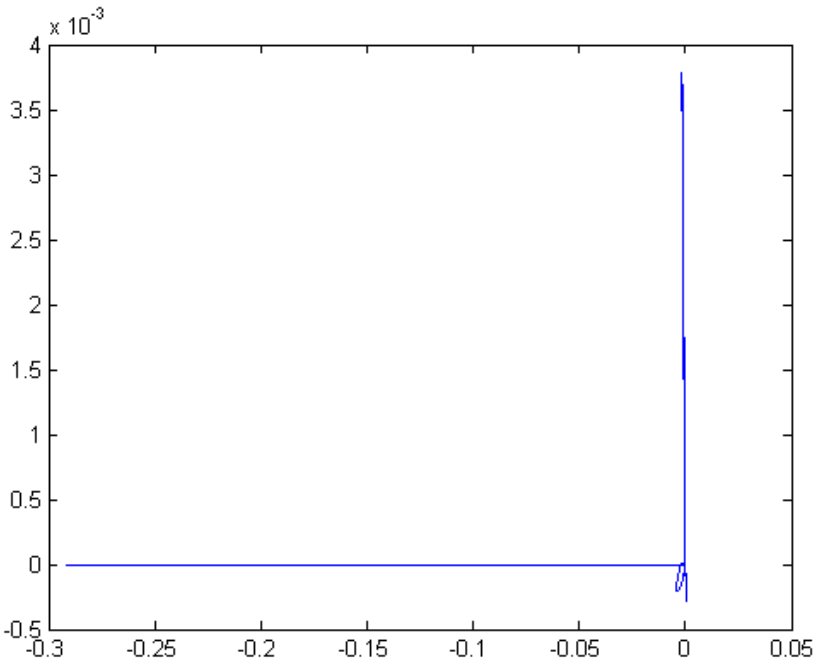
Simuleringsresultat for 5. grads bane er vist i figuren (5-10).



Figur (5-11) 5.gradsnaturlig splinebane

Tilstandene er plassert på riktig sted i forhold til referanse bane. Her ser vi at akselerasjon i det første baneelementet skissert som en parabel (ligner på det som vi har funnet i 4.grads bane).

Det er plottet en differanse mellom referanse bane og 5. grads bane som er vist nedenfor



Figur (5.12) differanse mellom referanse bane og 5.grads naturlig spline bane

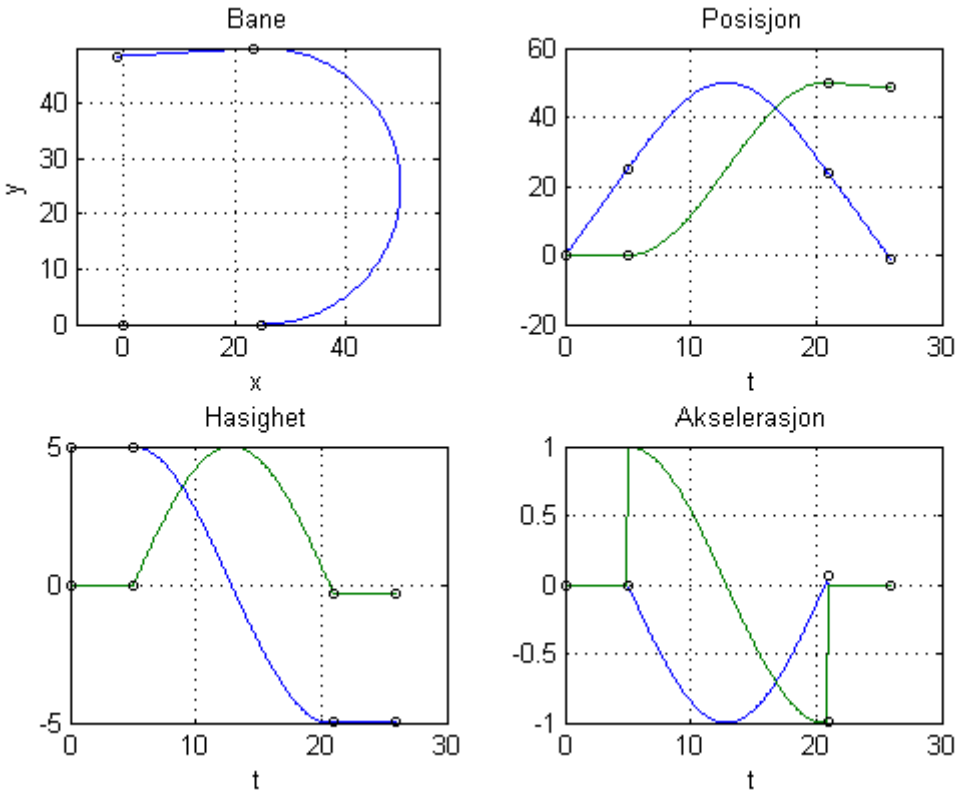
Feilen som oppstår er for lite som kan neglisjeres. Det kan konkluderes med at 5.grads naturlig spline algoritmen gir oss et tilfredsstillende resultat siden kravet for kontinuitet i hastighet og akselerasjon er oppfylt.

5.4 bane i sirkelbevegelse

- Referansebane

Her vil det undersøke om det er mulig å erstatte bane som representert av Eulerspiral og sirkelbue med 5.grads naturlig spline algoritmen. Det vil først visualisere en bane med figuren (5.13), som er bygd av 3 baneelementer rettlinje, sirkel og rettlinje.

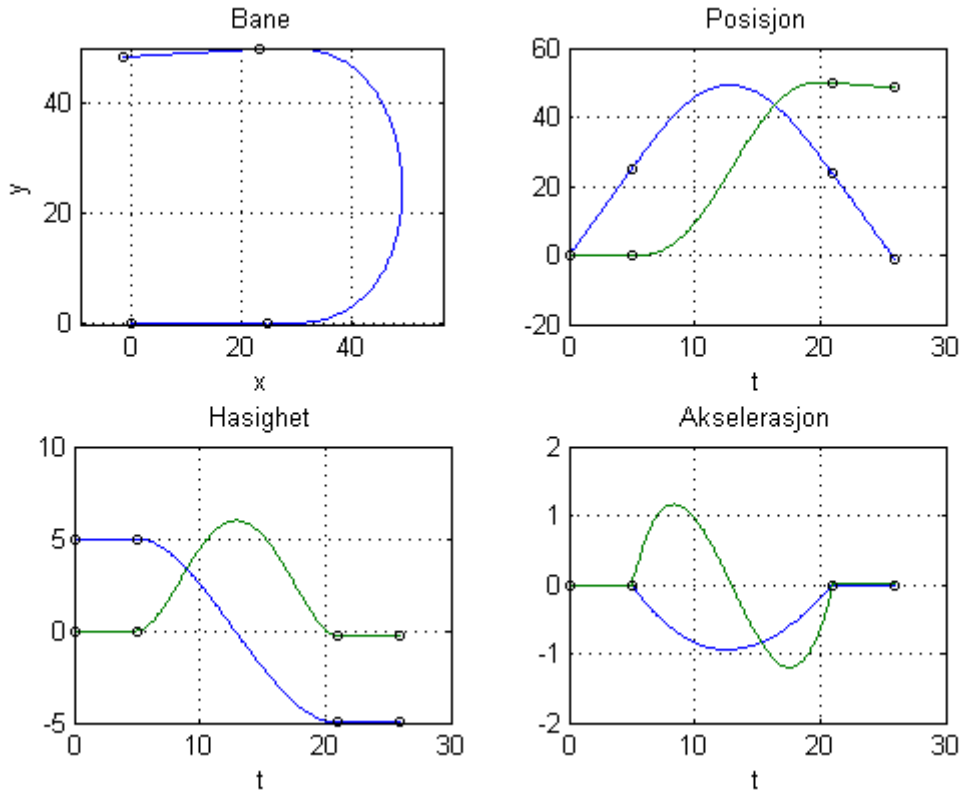
Det er beskrevet i starten at fartøyet vil ikke kjøre jevn fra rettlinje til en sirkel, derfor ble det brukt Eulerspiral i overgangen. Figuren skisserer en bane uten Eulerspiral. Feilen betraktes tydelig fra akselerasjonsplott hvor i første sving får x-retning en negativ påvirkning til å utforme en sinus funksjon men y- retning får en umiddelbar positiv påvirkning og reduseres gradvis ned så økes umiddelbar igjen og fortsetter lik null i det siste baneelementet.



Figur (5-13) bane i sirkelbevegelse

- 5.grads naturlig spline bane

Sirkelbane for 5.grads naturlig spline kurve, viser til simuleringsresultatet som følger



Figur (5.14) 5.gradsnaturlig spline bane i sirkelbevegelse

Plott av posisjon og hastighet stemmer overens i forhold til referanse bane. Akselerasjon er satt lik null i endepunkter for hvert baneelement for å oppfylle kravet for kontinuitet. Vi ser at akselerasjon tar en parabel form i svingen og er glatt som det skal i motsatt til referanse bane som er vist i figur(5.13) uten Eulerspiral. Det vil si at man kan erstatte Eulerspiral og sirkelbue med 5.grads naturlig spline algoritme.

6 Konklusjon og videre arbeid

6.1 Konklusjon

Målet for dette prosjektet er å finne en passende algoritme for banegenerator som gir ut kontinuerlig hastighet og akselerasjon.

Fremgangsmåte ved bruk av Eulerspiral og sirkel for å realisere en banegenerator, har vist å gi et godt resultat. Banen blir av ønsket form etter hva brukeren av programmet setter initialverdier, og fremstiller en visuell figur som gir brukeren en oversikt over hvordan banen blir i forhold til de betingelsene som er bestemt. Dermed kan banen brukes som en referanse bane.

Den 2-dimensjonale banen med tilhørende posisjon, hastighet og akselerasjon gir ut avbildning av hvordan baneelementer er koblet sammen med skjøtepunkter. Skjøtepunktene kan brukes videre som veipunkter til oppbygging av spline metode.

Det ble testet tre algoritmer for spline metode og sammenliknet resultatene med referanse bane. Utledning av matematiske likninger og beregninger for de forskjellige algoritmer har testet ut og vist at femte grads naturlig spline har resultert i forventede simuleringsresultater, og konkludert med aksept mål.

I en spline metode kan det oppdages en del numerisk feil, men feilen vil ikke påvirke andre baneelementer siden endepunkter, hastighet og akselerasjon i endepunkter har blitt spesifisert i referanse bane for hvert baneelement.

Oppgaven har vært veldig lærerik og arbeidet som er utført har gjort det mulig å få forskjellige måter å presentere banen med et tilfredsstillende resultat.

6.2 Videre arbeid

Videre arbeidet som kan være aktuelt,

- Utvide banegenerator på en enklere måte til å spesifisere veipunkter og tilhørende hastighet og akselerasjon.
- Finne vinkelhastighet og spesifikk kraft for et fly som kan fly rent.
- Det kan også finne vinkelhastighet, vinkelakselerasjon og spesifikk kraft for Gyro og akselerometer.

Referanser

[1] O. Hallingstad, *Matematisk modellering av dynamiske systemer,* 2008.

[2] K. M. Tom Lyche, Spline Methods, Oslo , 2011.

[3] R. F. G. E. Elaine Cohen, Geometric Modeling with Splines, 2001.

[4] D. H. House, «Spline curves,» June 2010. [Internett]. Available: http://people.cs.clemson.edu/~dhouse/courses/405/notes/splines.pdf.

[5] M.Lekkas, Andreas R. Dahl, Morten Breivik, Thor I.Fossen , «Continuous- Curvature Path Generation Using Fermat's Spline,» 2013.

Vedlegg

A. Funksjon

A.1 RBA.m

```
function [p__a,v__a,a__a]=RBA(t,T,V_0,V_1);
  A=(V_1-V_0)/T;  om=2*pi/T; N=size(t,2);
  p__a=[V_0*t+A*t.^2/2-A*(1-cos(om*t))/om^2;zeros(1,N)];
  v__a=[V_0+A*(t-sin(om*t)/om);zeros(1,N)];
  a__a=[A*(1-cos(om*t));zeros(1,N)];
end
```

A.2 SB.m

```
function [p__a,v__a,a__a]=SB(t,T,V,A)
%  Sirkelbaneelement
  om=A/V; th=om*t; R=V^2/A;
  p__a=R*[sin(th);1-cos(th)];
  v__a=V*[cos(th);sin(th)];
  a__a=A*[-sin(th);cos(th)];
end
```

A.3 ES.m

```
function [p__a,v__a,a__a]=ES(t,T,V,A)
  th=(A/(2*V*T))*t.^2;  Om=sqrt(A/(pi*V*T)); N=size(t,2);
  p__a=(V/Om)*[fresnelc(Om*t);fresnels(Om*t)];
  v__a=V*[cos(th);sin(th)];
  a__a=(A/T)*[-t.*sin(th);t.*cos(th)];
end
```

A.4 OES.m

```
function [p__a,v__a,a__a]=OES(t,T,V,A)
  th=(A/V)*(t-t.^2/(2*T)); th_T=A*T/(2*V);
  Om=sqrt(A/(pi*V*T)); N=size(t,2);
  p__a=(V/Om)*[-cos(th_T),-sin(th_T);-sin(th_T),cos(th_T)]*[fresnelc(Om*(T-
t))-fresnelc(Om*T);fresnels(Om*(T-t))-fresnels(Om*T)];
  v__a=V*[cos(th);sin(th)];
  a__a=(A/T)*[-(T-t).*sin(th);(T-t).*cos(th)];
end
```

A.5 PB.m

```
function [p__a,v__a,a__a]=PB(t,t1,t2,p1,p2,v1,v2,a1,a2)
  P=@(x) [x^5 x^4 x^3 x^2 x 1];
  V=@(x) [5*x^4 4*x^3 3*x^2 2*x 1 0];
  A=@(x) [20*x^3 12*x^2 6*x 2 0 0];
  Oz=zeros(1,6);
  P1=P(t1);  P2=P(t2);
  V1=V(t1);  V2=V(t2);
  A1=A(t1);  A2=A(t2);
```

```
  B=[P1 Oz; Oz P1; P2 Oz; Oz P2;...
     V1 Oz; Oz V1; V2 Oz; Oz V2;...
     A1 Oz; Oz A1; A2 Oz; Oz A2];
  c=[p1; p2; v1; v2; a1; a2];
  a=zeros(6,1); b=zeros(6,1);
  d=inv(B)*c; a=d(1:6,1);b=d(7:12,1);
  N=size(t,2);
  p__a=zeros(2,N); v__a=zeros(2,N); a__a=zeros(2,N);
  for k=1:N
    p__a(:,k)=[P(t(k))*a;P(t(k))*b];
    v__a(:,k)=[V(t(k))*a;V(t(k))*b];
    a__a(:,k)=[A(t(k))*a;A(t(k))*b];
  end
end
```

A.6 SP3B.m

```
function [p__n,v__n,a__n]=SP3B(t,twp,Pwp,v1,a1)
%  Splinebane
  N=size(twp,2)-1;            % Antall baneelementer
  P=@(x) [x^3 x^2 x 1];   Ph=zeros(N+1,4);
  V=@(x) [3*x^2 2*x 1 0]; Vh=zeros(N+1,4);
  A=@(x) [6*x 2 0 0];     Ah=zeros(N+1,4);
  for k=1:N+1
    Ph(k,:)=P(twp(1,k));
    Vh(k,:)=V(twp(1,k));
    Ah(k,:)=A(twp(1,k));
  end

  F=zeros(4*N,4*N);
  F(1,1:4)=Vh(1,:); F(2,1:4)=Ah(1,:);
  for k=1:N
    F(2*k+1,4*k-3:4*k)=Ph(k,1:4);
    F(2*k+2,4*k-3:4*k)=Ph(k+1,1:4);
  end
  iV=2+2*N;iA=iV+N-1;
  for k=1:N-1
    F(iV+k,4*k-3:4*k)=Vh(k+1,1:4);
    F(iV+k,4*k+1:4*k+4)=-Vh(k+1,1:4);
    F(iA+k,4*k-3:4*k)=Ah(k+1,1:4);
    F(iA+k,4*k+1:4*k+4)=-Ah(k+1,1:4);
  end

  h1=zeros(4*N,1); h2=zeros(4*N,1);
  h1(1,1)=v1(1); h1(2,1)=a1(1);
  h2(1,1)=v1(2); h2(2,1)=a1(2);
  for k=1:N
    h1(2*k+1,1)=Pwp(1,k)'; h1(2*k+2,1)=Pwp(1,k+1)';
    h2(2*k+1,1)=Pwp(2,k)'; h2(2*k+2,1)=Pwp(2,k+1)';
  end

  a=inv(F)*h1; b=inv(F)*h2;

 Nt=size(t,2);
  p__n=zeros(2,Nt);
  v__n=zeros(2,Nt);
  a__n=zeros(2,Nt);
```

```
  k=1;
  for i=1:N
    for j=1:twp(2,i)
      p__n(:,k)=[P(t(k))*a(4*(i-1)+1:4*i);P(t(k))*b(4*(i-1)+1:4*i)];
      v__n(:,k)=[V(t(k))*a(4*(i-1)+1:4*i);V(t(k))*b(4*(i-1)+1:4*i)];
      a__n(:,k)=[A(t(k))*a(4*(i-1)+1:4*i);A(t(k))*b(4*(i-1)+1:4*i)];
      k=k+1;
    end
    k=k-1;
  end
end
```

A.7 SP4B.m

```
function [p__n,v__n,a__n]=SP4B(t,twp,Pwp,Vwp,a1)
% 4.ordens splinetrajektor
  N=size(twp,2)-1;           % Antall baneelementer
  P=@(x) [x^4 x^3 x^2 x 1];     Ph=zeros(N+1,5);
  V=@(x) [4*x^3 3*x^2 2*x 1 0]; Vh=zeros(N+1,5);
  A=@(x) [12*x^2 6*x 2 0 0];    Ah=zeros(N+1,5);
  for k=1:N+1
    Ph(k,:)=P(twp(1,k));
    Vh(k,:)=V(twp(1,k));
    Ah(k,:)=A(twp(1,k));
  end

  F=zeros(5*N,5*N);
  F(1,1:5)=Ah(1,:);
  for k=1:N
    F(2*k,5*k-4:5*k)=Ph(k,1:5);
    F(2*k+1,5*k-4:5*k)=Ph(k+1,1:5);
  end
  iV=2*N;
  for k=1:N
    F(iV+2*k,5*k-4:5*k)=Vh(k,1:5);
    F(iV+2*k+1,5*k-4:5*k)=Vh(k+1,1:5);
  end
  iA=4*N+1;
  for k=1:N-1
    F(iA+k,5*k-4:5*k)=Ah(k+1,1:5);
    F(iA+k,5*k+1:5*k+5)=-Ah(k+1,1:5);
  end

  h1=zeros(5*N,1); h2=zeros(5*N,1);
  h1(1,1)=a1(1); h2(1,1)=a1(2);
  for k=1:N
    h1(2*k,1)=Pwp(1,k)'; h1(2*k+1,1)=Pwp(1,k+1)';
    h2(2*k,1)=Pwp(2,k)'; h2(2*k+1,1)=Pwp(2,k+1)';
    h1(iV+2*k,1)=Vwp(1,k)'; h1(iV+2*k+1,1)=Vwp(1,k+1)';
    h2(iV+2*k,1)=Vwp(2,k)'; h2(iV+2*k+1,1)=Vwp(2,k+1)';
  end

  a=inv(F)*h1; b=inv(F)*h2;

  Nt=size(t,2);
  p__n=zeros(2,Nt);
```

```
  v__n=zeros(2,Nt);
  a__n=zeros(2,Nt);
  k=1;
  for i=1:N
    for j=1:twp(2,i)
      p__n(:,k)=[P(t(k))*a(5*(i-1)+1:5*i);P(t(k))*b(5*(i-1)+1:5*i)];
      v__n(:,k)=[V(t(k))*a(5*(i-1)+1:5*i);V(t(k))*b(5*(i-1)+1:5*i)];
      a__n(:,k)=[A(t(k))*a(5*(i-1)+1:5*i);A(t(k))*b(5*(i-1)+1:5*i)];
      k=k+1;
    end
    k=k-1;
  end
end
```

A.8 SP5B.m

```
function [p__n,v__n,a__n]=SP5B(t,twp,Pwp,Vwp,Awp)
% 6.ordens splinetrajektor
  N=size(twp,2)-1;           % Antall baneelementer
  P=@(x) [x^5 x^4 x^3 x^2 x 1];       Ph=zeros(N+1,6);
  V=@(x) [5*x^4 4*x^3 3*x^2 2*x 1 0]; Vh=zeros(N+1,6);
  A=@(x) [20*x^3 12*x^2 6*x 2 0 0];   Ah=zeros(N+1,6);
  for k=1:N+1
    Ph(k,:)=P(twp(1,k));
    Vh(k,:)=V(twp(1,k));
    Ah(k,:)=A(twp(1,k));
  end

  F=zeros(6,6); a=zeros(N*6,1); b=a;
  for k=1:N
    F(1,1:6)=Ph(k,1:6); F(2,1:6)=Ph(k+1,1:6);
    F(3,1:6)=Vh(k,1:6); F(4,1:6)=Vh(k+1,1:6);
    F(5,1:6)=Ah(k,1:6); F(6,1:6)=Ah(k+1,1:6);
    h1=[Pwp(1,k);Pwp(1,k+1);Vwp(1,k);Vwp(1,k+1);Awp(1,k);Awp(1,k+1)];
    h2=[Pwp(2,k);Pwp(2,k+1);Vwp(2,k);Vwp(2,k+1);Awp(2,k);Awp(2,k+1)];
    ah=inv(F)*h1; bh=inv(F)*h2;
    a(6*k-5:6*k)=ah; b(6*k-5:6*k)=bh;
  end

  Nt=size(t,2);
  p__n=zeros(2,Nt);
  v__n=zeros(2,Nt);
  a__n=zeros(2,Nt);
  k=1;
  for i=1:N
    for j=1:twp(2,i)
      p__n(:,k)=[P(t(k))*a(6*(i-1)+1:6*i);P(t(k))*b(6*(i-1)+1:6*i)];
      v__n(:,k)=[V(t(k))*a(6*(i-1)+1:6*i);V(t(k))*b(6*(i-1)+1:6*i)];
      a__n(:,k)=[A(t(k))*a(6*(i-1)+1:6*i);A(t(k))*b(6*(i-1)+1:6*i)];
      k=k+1;
    end
    k=k-1;
  end
end
```

B. Simulering

```
% ------'Sammensattbane'---------
% Banedata-----------------------------------------
% Kolonne 1: banetype           Kolonne 2   3   4
%   1 - rett strekning                    T V_0 V_1
%   2 - Eulerspiral                       T   V   A
%   3 - Sirkel                            T   V   A
%   4 - Omvendt Eulerspiral               T   V   A
BD=[1  5  2  5; ...
     1  5  5  5;....
     2  5  5  1; ...
     3  8  5  1; ...
     4  5  5  1; ...
     1  5  5  5 ] ;

  NB=size(BD,1);        % Antall baneelementer
  p_k_n=zeros(2,NB+1);
  th=zeros(1,NB+1);
  p_k_n(:,1)=[0;0];     % Startposisjon for baneelementet
  th(1)=0;              % Startvinkel for banen
  dt=0.1 ;              % Tidsinkrement
% Generer banen-------------------------------------
% Banelementa ligger etterhverandre med start og sluttverdier
% slik at tider og verdier gjentas for sammenknytningspunkta.
for k=1:NB
  T=BD(k,2);V=BD(k,3); AV=BD(k,4);
  t=0:dt:T; NT=size(t,2);
  if BD(k,1)==1
    [p__a,v__a,a__a]=RBA(t,T,V,AV);
    th_v=0;
  elseif BD(k,1)==2
    [p__a,v__a,a__a]=ES(t,T,V,AV);
    th_v=AV*T/(2*V);
  elseif BD(k,1)==3
    [p__a,v__a,a__a]=SB(t,T,V,AV);
    th_v=+AV*T/V;
  elseif BD(k,1)==4
    [p__a,v__a,a__a]=OES(t,T,V,AV);
    th_v=AV*T/(2*V);
  else
    print('Feil banetype')
  end
  th(k+1)=th(k)+th_v;
  p_k_n(:,k+1)=p_k_n(:,k)+R_3(th(k))*p__a(:,NT);  %Posisjon ved enden av
baneelement k+1
  if k==1
    tp=[t];
    p__n=repmat(p_k_n(:,k),1,NT)+R_3(th(k))*p__a(:,:);
    v__n=R_3(th(k))*v__a;
    a__n=R_3(th(k))*a__a;
    twp=[tp(1); NT];
    twp=[twp [tp(1,end);0]];
    Pwp=p__n(:,1);
    Pwp=[Pwp p__n(:,end)];
    Vwp=v__n(:,1);
    Vwp=[Vwp v__n(:,end)];
    Awp=a__n(:,1);
    Awp=[Awp a__n(:,end)];
  else
    tp=[tp(1,1:end-1) tp(end)+t];
```

```
    p__n=[p__n(:,1:end-1),repmat(p_k_n(:,k),1,NT)+R_3(th(k))*p__a(:,:)];
    v__n=[v__n(:,1:end-1),R_3(th(k))*v__a];
    a__n=[a__n(:,1:end-1), R_3(th(k))*a__a];
    twp(2,k)=NT;
    twp=[twp [tp(end);0]];
    Pwp=[Pwp p__n(:,end)];
    Vwp=[Vwp v__n(:,end)];
    Awp=[Awp a__n(:,end)];
  end;
end


% Plotting av sammensatt bane
figure(1);
subplot(2,2,1); plot(p__n(1,:),p__n(2,:)); hold on;

scatter(Pwp(1,:),Pwp(2,:),20,'k')
grid on; axis equal;
title('Bane'); xlabel('x'); ylabel('y')

subplot(2,2,2); plot(tp,p__n); hold on;
scatter(twp(1,:),Pwp(1,:),20,'k')
scatter(twp(1,:),Pwp(2,:),20,'k')
grid on; title('Posisjon'); xlabel('t');

subplot(2,2,3); plot(tp,v__n); hold on;
scatter(twp(1,:),Vwp(1,:),20,'k')
scatter(twp(1,:),Vwp(2,:),20,'k')
grid on; title('Hasighet'); xlabel('t'); hold;

subplot(2,2,4); plot(tp,a__n); hold on;
scatter(twp(1,:),Awp(1,:),20,'k')
scatter(twp(1,:),Awp(2,:),20,'k')
grid on; title('Akselerasjon'); xlabel('t')

%-------------------------------------------------------------
% 3. ordens splinebane
%-------------------------------------------------------------
v1=Vwp(:,1); a1=Awp(:,1);
[p_sp3_n, v_sp3_n, a_sp3_n]=SP3B(tp,twp,Pwp,v1,a1);

figure(2);
subplot(2,2,1); plot(p_sp3_n(1,:),p_sp3_n(2,:)); hold on;
scatter(Pwp(1,:),Pwp(2,:),20,'k')
grid on; axis equal;
title('Bane'); xlabel('x'); ylabel('y')

subplot(2,2,2); plot(tp,p_sp3_n); hold on;
scatter(twp(1,:),Pwp(1,:),20,'k')
scatter(twp(1,:),Pwp(2,:),20,'k')
grid on; title('Posisjon'); xlabel('t');

subplot(2,2,3); plot(tp,v_sp3_n); hold on;
scatter(twp(1,:),Vwp(1,:),20,'k')
scatter(twp(1,:),Vwp(2,:),20,'k')
grid on; title('Hasighet'); xlabel('t'); hold;

subplot(2,2,4); plot(tp,a_sp3_n); hold on;
scatter(twp(1,:),Awp(1,:),20,'k')
```

```
scatter(twp(1,:),Awp(2,:),20,'k')
grid on; title('Akselerasjon'); xlabel('t')

%-------------------------------------------------------------
% 4. ordens splinebane
%-------------------------------------------------------------
a1=Awp(:,1);
[p_sp4_n, v_sp4_n, a_sp4_n]=SP4B(tp,twp,Pwp,Vwp,a1);

figure(3);
 subplot(2,2,1); plot(p_sp4_n(1,:),p_sp4_n(2,:)); hold on;
scatter(Pwp(1,:),Pwp(2,:),20,'k')
grid on; axis equal;
title('Bane'); xlabel('x'); ylabel('y')

subplot(2,2,2); plot(tp,p_sp4_n); hold on;
scatter(twp(1,:),Pwp(1,:),20,'k')
scatter(twp(1,:),Pwp(2,:),20,'k')
grid on; title('Posisjon'); xlabel('t');

subplot(2,2,3); plot(tp,v_sp4_n); hold on;
scatter(twp(1,:),Vwp(1,:),20,'k')
scatter(twp(1,:),Vwp(2,:),20,'k')
grid on; title('Hasighet'); xlabel('t'); hold;

subplot(2,2,4); plot(tp,a_sp4_n); hold on;
scatter(twp(1,:),Awp(1,:),20,'k')
scatter(twp(1,:),Awp(2,:),20,'k')
grid on; title('Akselerasjon'); xlabel('t')

%-------------------------------------------------------------
% 5. ordens splinebane
%-------------------------------------------------------------
 [p_sp5_n, v_sp5_n, a_sp5_n]=SP5B(tp,twp,Pwp,Vwp,Awp);

figure(4);
subplot(2,2,1); plot(p_sp5_n(1,:),p_sp5_n(2,:)); hold on;
scatter(Pwp(1,:),Pwp(2,:),20,'k')
grid on; axis equal;
title('Bane'); xlabel('x'); ylabel('y')

subplot(2,2,2); plot(tp,p_sp5_n); hold on;
scatter(twp(1,:),Pwp(1,:),20,'k')
scatter(twp(1,:),Pwp(2,:),20,'k')
grid on; title('Posisjon'); xlabel('t');

subplot(2,2,3); plot(tp,v_sp5_n); hold on;
scatter(twp(1,:),Vwp(1,:),20,'k')
scatter(twp(1,:),Vwp(2,:),20,'k')
grid on; title('Hasighet'); xlabel('t'); hold;

subplot(2,2,4); plot(tp,a_sp5_n); hold on;
scatter(twp(1,:),Awp(1,:),20,'k')
scatter(twp(1,:),Awp(2,:),20,'k')
grid on; title('Akselerasjon'); xlabel('t')
```